МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ
имени М.В.ЛОМОНОСОВА

МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

АКАДЕМИЯ ТЕХНОЛОГИЧЕСКИХ НАУК РФ
РОССИЙСКАЯ АКАДЕМИЯ ЕСТЕСТВЕННЫХ НАУК
СОВЕТ ПО КИБЕРНЕТИКЕ РАН
МОСКОВСКИЙ НАУЧНЫЙ ЦЕНТР ПО КУЛЬТУРЕ И
ИНФОРМАЦИОННЫМ ТЕХНОЛОГИЯМ

# МАТЕРИАЛЫ
## X Международной конференции
## “Интеллектуальные системы и компьютерные науки”

(5–10 декабря 2011 года)

2011

УДК 519.95, 519.14, 519.1, 519.6

**Материалы X Международной конференции "Интеллектуальные системы и компьютерные науки" (5–10 декабря 2011 года).**

Сборник содержит работы участников X Международной конференции "Интеллектуальные системы и компьютерные науки", проходившей на механико-математическом факультете МГУ имени М.В.Ломоносова с 5 по 10 декабря 2011 года. Сборник адресован научным сотрудникам, преподавателям, аспирантам и студентам, работающим в области математических проблем теории интеллектуальных стстем и их приложений.

Научное издание

МАТЕРИАЛЫ X МЕЖДУНАРОДНОЙ КОНФЕРЕНЦИИ "ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ И КОМПЬЮТЕРНЫЕ НАУКИ" (5–10 декабря 2011 года) под общей редакцией академика Садовничего В.А., проф. Кудрявцева В.Б., проф. Михалева А.В.

В составлении и редактировании принимали участие: Алексеев Д.В., Галатенко А.В., Гасанов Э.Э., Строгалов А.С., Шуткин Ю.С.

Ответственный за выпуск Строгалов А.С.

# Оглавление

# Секция “Интеллектуальные системы”

# О ВОССТАНОВЛЕНИИ ИЗОБРАЖЕНИЙ ПО КОДАМ В НЕКОТОРЫХ ВЫРОЖДЕННЫХ СЛУЧАЯХ

**Агниашвили П.Г. (МГУ им. М.В. Ломоносова)**
*colllapse@mail.ru*

Одной из ключевых характеристик изображения является его код. По своему смыслу код должен отражать определенную общность в восприятии изображений. В рассматриваемом дискретно-геометрическом подходе общим признаком является аффинная эквивалентность изображений. Данное направление получило развитие в книге [1], где, в частности, исследуется возможность восстановления изображений по их кодам в двумерном и трехмерном случаях.

В предлагаемой работе исследуется аналогичная проблема в общем случае произвольной конечной размерности. Отдельное внимание уделяется вырожденному случаю изображений, лежащих в двух параллельных гиперплоскостях.

Всюду далее рассматривается пространство $\mathbb{R}^n$, $n \geq 2$. Под точкой с индексом $p \in \mathbb{N}$ будем понимать упорядоченную пару $(\mathbf{x}, p)$, где $\mathbf{x} = (x_1, \ldots, x_n)$ – вектор из $\mathbb{R}^n$.

*Изображением первого рода* в $\mathbb{R}^n$ называется конечное множество индексированных точек, не лежащих в одной гиперплоскости и занумерованных индексами $1, \ldots, k$ в случае $k$ точек, $k \in \mathbb{N}$.

Для изображения первого рода $A$ через $|A|$ будем обозначать число точек в изображении, а через $\mathbb{N}_{|A|} = \{1, \ldots, |A|\}$ – множество индексов у точек изображения. Здесь и далее запись $\mathbb{N}_k$ используется для обозначения начального отрезка натурального ряда.

Два изображения первого рода $A_1$ и $A_2$, состоящие из одинакового числа точек ($|A_1| = |A_2|$), называются *a-эквивалентными*, если существует аффинное преобразование, при котором каждая точка из $A_1$ с индексом $p \in \mathbb{N}_{|A_1|}$ отображается в точку из $A_2$ с тем же индексом $p \in \mathbb{N}_{|A_2|}$.

*Изображением второго рода* называется класс всех попарно a-эквивалентных изображений первого рода.

Далее под изображением понимается изображение первого или второго рода. Род изображения не уточняется в тех случаях, когда исследуемые свойства сохраняются при аффинных преобразованиях.

Рассмотрим изображение $A$. Введем произвольную аффинную систему координат в $\mathbb{R}^n$ и пусть $(x_1^p, \ldots, x_n^p)$ – координаты точки с

индексом $p \in \mathbb{N}_{|A|}$ в этой системе координат. Для произвольных индексов $r_1, \ldots, r_{n+1}, s_1, \ldots, s_{n+1} \in \mathbb{N}_{|A|}$ определим индексированное число $\mu_{s_1 \ldots s_{n+1}}^{r_1 \ldots r_{n+1}}$ по формуле:

$$\mu_{s_1 \ldots s_{n+1}}^{r_1 \ldots r_{n+1}} = \begin{vmatrix} x_1^{r_1} & \cdots & x_n^{r_1} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1^{r_{n+1}} & \cdots & x_n^{r_{n+1}} & 1 \end{vmatrix} \Bigg/ \begin{vmatrix} x_1^{s_1} & \cdots & x_n^{s_1} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1^{s_{n+1}} & \cdots & x_n^{s_{n+1}} & 1 \end{vmatrix}$$

В случае равенства знаменателя нулю полагаем, что значение $\mu_{s_1 \ldots s_{n+1}}^{r_1 \ldots r_{n+1}}$ не определено, и обозначаем такой случай знаком $\infty$. В случае ненулевых определителей индексированное число является отношением ориентированных объемов $n$-симплексов на соответствующих точках.

*М-кодом* изображения $A$ называется множество всех индексированных чисел $\mu\text{-}T_A = \{\mu_{s_1 \ldots s_{n+1}}^{r_1 \ldots r_{n+1}} | r_i, s_j \in \mathbb{N}_{|A|}\}$.

*Симплексным набором* индексов для м-кода $\mu\text{-}T_A$ называется набор $S = (i_1, \ldots, i_{n+1})$, для которого $\mu_{\mathbf{i_1}, \ldots, \mathbf{i_{n+1}}}^{r_1, \ldots, r_{n+1}} \neq \infty$ при любых $r_i \in \mathbb{N}_{|A|}$. Другими словами, точки с индексами $i_1, \ldots, i_{n+1}$ не лежат в одной гиперплоскости, на что и указывает название набора.

Обозначим через $\mu_j^s$ элемент м-кода, у которого нижний набор является симплексным набором $(i_1, \ldots, i_{n+1})$, а верхний набор отличается от симплексного в $j$-ой позиции: $(i_1, \ldots, i_{j-1}, s, i_{j+1}, \ldots, i_{n+1})$.

*Ключевым подмножеством* м-кода $\mu\text{-}T_A$ относительно симплексного набора $S$ называется множество $\{\mu_j^s | s \in \mathbb{N}_{|A|}, j \in \mathbb{N}_{n+1}\}$.

Имеют место следующие соотношения:

$$\mu_1^s + \ldots + \mu_{n+1}^s = 1 \tag{1}$$

$$\mu_{s_1 \ldots s_{n+1}}^{r_1 \ldots r_{n+1}} = \begin{vmatrix} \mu_1^{r_1} & \cdots & \mu_{n+1}^{r_1} \\ \vdots & \ddots & \vdots \\ \mu_1^{r_{n+1}} & \cdots & \mu_{n+1}^{r_{n+1}} \end{vmatrix} \Bigg/ \begin{vmatrix} \mu_1^{s_1} & \cdots & \mu_{n+1}^{s_1} \\ \vdots & \ddots & \vdots \\ \mu_1^{s_{n+1}} & \cdots & \mu_{n+1}^{s_{n+1}} \end{vmatrix} \tag{2}$$

Данные соотношения используются, в частности, при доказательстве теорем 1 и 2.

**Теорема 1.** *Между множеством м-кодов и множеством изображений второго рода существует биекция, сопоставляющая каждому изображению его м-код.*

*Кодом* изображения $A$ называется множество всех индексированных чисел $T_A = \{\rho_{s_1\ldots s_{n+1}}^{r_1\ldots r_{n+1}} | r_i, s_j \in \mathbb{N}_{|A|}\}$, где $\rho_{s_1\ldots s_{n+1}}^{r_1\ldots r_{n+1}} = |\mu_{s_1\ldots s_{n+1}}^{r_1\ldots r_{n+1}}|$. Данное определение согласуется и обобщает определение кода из [1].

Рассмотрим два изображения $A$ и $\hat{A}$ со следующими свойствами: $|A| = |\hat{A}|$, $S = (i_1, \ldots, i_{n+1})$ – общий симплексный набор, и ключевые подмножества м-кодов равны по модулю ($|\mu_j^s| = |\hat{\mu}_j^s|$). Положим $M_+^s = \{i_j \in S | \mu_j^s = \hat{\mu}_j^s \neq 0\}$ и $M_-^s = \{i_j \in S | \mu_j^s = -\hat{\mu}_j^s \neq 0\}$.

*Разделяющим разбиением* набора $S$ называется пара множеств $(S_1, S_2)$, для которых выполняется: $S = S_1 \sqcup S_2$ и для любого индекса $s \in \mathbb{N}_{|A|}$ либо $M_+^s \subseteq S_1, M_-^s \subseteq S_2$, либо $M_+^s \subseteq S_2, M_-^s \subseteq S_1$.

Здесь и далее $S = S_1 \sqcup S_2$ означает $S = S_1 \cup S_2$ и $S_1 \cap S_2 = \varnothing$.

**Теорема 2.** *Коды изображений $A$ и $\hat{A}$ с перечисленными свойствами совпадают тогда и только тогда, когда существует разделяющее разбиение симплексного набора $S$.*

Рассмотрим произвольную аффинную систему координат в $\mathbb{R}^n$ и пусть $M = \{\mathbf{x}^{s_1}, \ldots, \mathbf{x}^{s_m}\}$ – некоторое множество точек.

*Гранью* $<M>$, порожденной множеством $M$, называется множество $<M> = \{\alpha_1\mathbf{x}^{s_1} + \ldots + \alpha_m\mathbf{x}^{s_m} | \sum_{i=1}^m \alpha_i = 1\}$.

Любая грань $<M>$ является аффинным подпространством, т.е. сдвигом соответствующего линейного подпространства $L$ на некоторый вектор $\mathbf{x} \in <M>$.

*Размерностью* $\dim(M)$ грани $<M>$ называется размерность соответствующего линейного подпространства $L$.

Изображение $A$ называется *допустимым*, если для любых двух непересекающихся граней $<M_1>$ и $<M_2>$, содержащих все точки изображения $A$, выполняется: $\dim(M_1) + \dim(M_2) = \dim(A) - 1$.

**Теорема 3.** *Коду соответствует единственное изображение второго рода тогда и только тогда, когда изображение является допустимым.*

Теорема 3 является аналогом утверждений, доказанных в [1] для случаев $\mathbb{R}^2$ и $\mathbb{R}^3$, а также общего результата, полученного Руденко А.Д. для случая пространства произвольной конечной размерности. Согласно этим результатам, коду соответствует единственное изображение с точностью до а-эквивалентности, если оно не лежит в двух параллельных гиперплоскостях. Теорема 3 применима и к тако-

му вырожденному случаю. Следующее утверждение более детально разбирает данный вопрос.

**Утверждение.** *Пусть* $A = A_1 \sqcup A_2$, *и* $< A_1 > \cap < A_2 > = \varnothing$. *Изображение* $A$ *допустимо* $\Leftrightarrow \dim(A_1) + \dim(A_2) = \dim(A) - 1$ , *и* $A_1$, $A_2$ *допустимы.*

Используем полученный результат для классификации допустимых изображений, лежащих в двух параллельных гиперплоскостях, в случаях $\dim(A) = 2$ и $\dim(A) = 3$.

**Случай $\dim(A) = 2$.** В данном случае изображение лежит на двух параллельных прямых. Все точки расположены на прямой и в точке, не лежащей на этой прямой.

**Случай $\dim(A) = 3$.** В данном случае изображение лежит на двух параллельных плоскостях. Возможно два варианта:

1. Все точки расположены на плоскости и в точке, не лежащей на этой плоскости. При этом точки на плоскости не могут быть расположены на двух параллельных прямых.

2. Все точки расположены на двух скрещивающихся прямых.

Автор выражает благодарность профессору Козлову Вадиму Никитовичу за постановку задачи и научное руководство, а также Алексееву Дмитрию Владимировичу за ценные замечания.

**Литература**

1. Козлов В. Н. Элементы математической теории зрительного восприятия. – М.: Издательство Центра прикладных исследований при механико-математическом факультете МГУ, 2001.

2. Михалев А. А., Михалев А. В. Начала алгебры, часть I. – М.: Интернет-университет информационных технологий, 2005.

3. Клейн Ф. Элементарная математика с точки зрения высшей, том 2, геометрия. – М.: Наука, 1987.

## ИСПОЛЬЗОВАНИЕ МЕТОДА В. Н. КОЗЛОВА В ОБРАЗОВАТЕЛЬНОМ ПРОЦЕССЕ НА КАФЕДРЕ МАТИС

**Алексеев Д.В. (МГУ, мех.–мат. ф–т)**
*dvalex@rambler.ru*

В работе рассматривается упрощенный вариант алгоритма В. Н. Козлова для восстановления трехмерных изображений по их плоским проекциям. Также показано использование упрощенного алгоритма В. Н. Козлова в образовательном процессе на кафедре МАТИС

**1. Введение** В работе используются следующие обозначения: точки на плоскости и в пространстве обозначаются прописными буквами: $A$, $B$, $C$,... Их координаты на плоскости и в пространстве обозначаются строчными буквами: $(a_x, a_y, a_z)$, $(b_x, b_y)$, ... Прямая, проходящая через точки $A$ и $B$ обозначается $(AB)$, плоскость, проведенная через точки $A$, $B$ и $C$ обозначается $\pi(ABC)$.

Алгоритм В. Н. Козлова восстановления трехмерного тела по плоским проекциям описан в работе [1]. Суть его такова — даны две проекции трехмерного тела (под телом понимается конечное множество точек в пространстве). Требуется восстановить исходное тело с точностью до аффинной эквивалентности, т.е. построить множество точек, которое было бы аффинно эквивалентно исходному.

Краткое изложение алгоритма приводится для основного случая.

Пусть $A'$, $B'$, $C'$, $D'$ и $E'$ — проекции точек $A$, $B$, $C$, $D$ и $E$ на плоскость $\pi_1$, а $A''$, $B''$, $C''$, $D''$ и $E''$ — на плоскость $\pi_2$.

**Определение ([1])** Рассмотрим на $\pi_1$ и $\pi_2$ четверку троек точек $((A'B'C')(A''B''C'')(C'E'D')(C''E''D''))$ такую, что $A'B'C'$ и $A''B''C''$ — треугольники, точки $D'$ и $E'$ лежат вне плоскости треугольника $A'B'C'$ (и, соответственно, точки $D''$ и $E''$ лежат вне плоскости треугольника $A''B''C''$) и из троек $C'D'E'$ и $C''D''E''$ хотя бы одна является треугольником. Такую четверку будем называть *правильной*. Определение того, что точка лежит в плоскости треугольника приведено в [1].

Пусть проекции точек $A$, $B$, $C$, $D$, $E$ образуют правильную четверку. Тогда, очевидно, существует единственное аффинное отображение, переводящее точки $A''$ , $B''$ и $C''$ в точки $A'$, $B'$, $C'$. Пусть при этом отображении точки $D''$ и $E''$ переходят в точки $\tilde{D}$ и $\tilde{E}$. Пусть $F'$ — точка пересечения прямых $D'E'$ и $\tilde{D}\tilde{E}$. Тогда прямая $CF$ являет-

ся проекцией прямой $l = \pi(ABC) \cap \pi(CDE)$. Следовательно, можно рассмотреть произвольные (не лежащие в одной плоскости) точки $A_0$, $B_0$, $C_0$ и $D_0$ в пространстве, построить прямую $l_0$, соответствующую прямой $C'F'$ и восстановить положение точки $E$.

**2. Упрощенная версия алгоритма В. Н. Козлова.** Упрощение алгоритма было предложено в процессе преподавания этой темы на 3 курсе мех.–мат. ф–та. Не ограничивая общности рассуждений можно считать, что точки $A'$, $B'$ и $C'$ обладают координатами $(a'_x, a'_y) = (1, 0)$, $(b'_x, b'_y) = (0, 1)$ и $c'_x, c'_y) = (0, 0)$, Действительно, всегда можно преобразовать эти точки с помощью некоторого аффинного преобразования, или сразу рассматривать аффинную систему координат с началом в точке $C'$ и базисными векторами $C'A'$ и $C'B'$.

Существует единственное аффинное преобразование, переводящее треугольник $A''B''C''$ в треугольник $A'B'C'$. Пусть при этом отображении точки $D''$ и $E''$ переходят в точки $\tilde{D}$ и $\tilde{E}$. Пусть $F'$ — точка пересечения прямых $D'E'$ и $\tilde{D}\tilde{E}$. Обозначим $\lambda = \frac{\tilde{E}F'}{\tilde{D}F'}$. Рассмотрим точки в пространстве $A_0(1, 0, 0)$, $B_0(0, 1, 0)$, $C_0(0, 0, 0)$, $D_0(d'_x, d'_y, 1)$ и $E_0(e'_x, e'_y, \lambda)$. Для указанных точек верна следующая

**Теорема 1.** Существуют направления проекции, такие, что проекции точек $A_0$, $B_0$, $C_0$, $D_0$ и $E_0$ на плоскость $z = 0$ аффинно эквивалентны проекциям $A', B', C', D', E'$ и $A'', B'', C'', D'', E''$.

**Доказательство.**

Выберем базис следующим образом : начало координат в точке $C'$, оси $x$ и $y$ проходят через точки $A'$ и $B'$ ось $Z$ перпендикулярна этой плоскости. Очевидно, что проекцией точек на эту плоскость по направлению, параллельному оси $Z$, являются точки $A', B', C', D', E'$.

Рассмотрим проекцию точек $A_0, B_0, C_0, D_0, E_0$ параллельно прямой $(E_0\tilde{E})$ (см. Рис. 1). Очевидно, проекцией точки $E_0$ будет точка $E$. Поскольку точки $D'E'F'$ являются проекциями пространственных точек $D$, $E$, $F$, а отношения отрезков при параллельном проектировании сохраняются, то $\frac{E'F'}{D'F'} = \frac{EF}{DF}$. Аналогично $\frac{E''F''}{D''F''} = \frac{EF}{DF}$, и, следовательно, $\frac{\tilde{E}F'}{\tilde{D}F'} = \frac{EF}{DF}$, поскольку аффинные преобразования сохраняют отношения отрезков на прямой. Из вышеуказанных пропорций вытекает подобие треугольников $\triangle D'\tilde{D}F' \simeq \triangle E'\tilde{E}F'$. Следовательно, отрезки $\tilde{D}D'$ и $\tilde{E}E'$ лежат на параллельных прямых и их отношение равно $\frac{\tilde{E}E'}{\tilde{D}D'} = \frac{\tilde{E}F'}{\tilde{D}F'} = \lambda$. Заметим, что из выбора точек

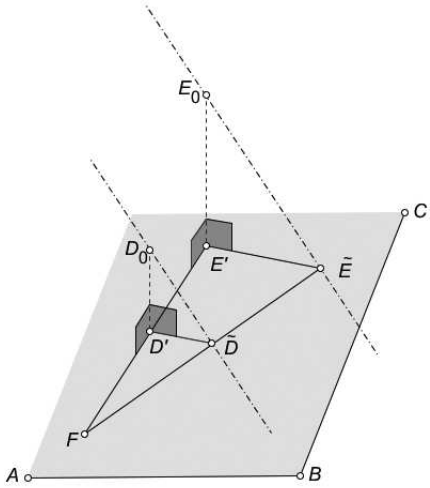



Рис. 1: Восстановление трехмерного изображения

$D_0$ и $E_0$ вытекает, что отношение $\frac{E_0E'}{D_0D'} = \lambda$. Следовательно, треугольники $\triangle D_0D'\tilde{D}$ и $\triangle E_0E'\tilde{E}$ подобны. Поскольку сторона $E_0E'$ параллельна $D_0D'$ по построению, а $E'\tilde{E}$ параллельна $D'\tilde{D}$ из подобия треугольников $\triangle D'\tilde{D}F'$ и $\triangle E'\tilde{E}F'$, то стороны $E_0\tilde{E}$ и $D_0\tilde{D}$ тоже будут параллельны. Следовательно, проекцией точки $D_0$ по направлению $E_0\tilde{E}$ на плоскость $\pi(A'B'C')$ будет точка $\tilde{D}$, что и требовалось доказать.

**3. Программа для автоматической генерации задач.** Для генерации задач с "хорошими"(т.е. целочисленными) ответами была написана программа на языке MATLAB, листинг приводится:

```
function f = generate_probleb()
%% Points  A,B,C are fixed , others --- arbitrary
 A0 = [1; 0], B0 = [0; 1], C0 = [0; 0], f=0;
 X0 = [2; 2]; % The pivot point
 D01 = [4; 0] , D02 = [2; -2]; % Arbitrary points
 factor = -0.5; % EX:DX ratio
 Transform1 = [ [ -1 3 1 ] ; [ 1 1 0 ] ];
```

```
 Transform2 = [ [ 2 -1 -3 ] ; [ -1 1 2 ] ];
    % Arbitrary transform matrixes
 E01 = X0 + factor * (D01 - X0), E02 = X0 + factor * (
     D02 - X0);
 names = { 'a'; 'b'; 'c'; 'd'; 'e'; 'x'
     };
 points1 = { A0; B0; C0; D01; E01; X0};
 points2 = { A0; B0; C0; D02; E02; X0};
NUM_POINTS = 6;
 points1tr= cell(NUM_POINTS,1), points2tr= cell(
     NUM_POINTS,1);
 for i =1:NUM_POINTS
    points1tr{i} = AffTrans(points1{i}, Transform1);
    points2tr{i} = AffTrans(points2{i}, Transform2);
 end
% Printing problem and answer
 fprintf(1,'Условие:\n\nпроекции на $\\Pi''$:␣\n');
 Disp(1,names, points1tr,NUM_POINTS-1, '''');
 fprintf('\n\nи проекции на $\\Pi''''$:␣\n');
 Disp(1,names, points2tr,NUM_POINTS-1, '''''');
 fprintf(1, '\n==========\nОтветы:\n');
 Disp(1,{names{6}}, {points1tr{6}}, 1, '''');% position
     X'
 Disp(1,{names{6}}, {points2tr{6}}, 1, '''''');%
     position X''
 fprintf(1, '␣\nуравнение прямой $L''$\n');
 PrintLineEquation(1, points1tr{3}, points1tr{6});% line
     C'X'
 fprintf(1, '␣\nуравнение прямой $L''''$\n');
 PrintLineEquation(1, points2tr{3}, points2tr{6});% line
     C''X''
 z = [ 0; 0; 0; 1; factor; 0];
 fprintf('\n\n␣Трехмерные координаты \n');
 Disp3(1, names, points1, z, NUM_POINTS - 1, '␣');
 D3=AffTrans(D01,Transform2), E3=AffTrans(E01,Transform2
     );
end
% some useful functions (headers only):
function Xt = AffTrans(X, Matr) % Makes affine transform
     with point X
function d = Disp3(hFile, names, coords, z, N, suffix)
```

```
    %prints points
function str = number2string(n) % Pretty printer for
    rational numbers
function f = PrintLineEquation(hFile, A, B) % Pretty
    printer for line equations
function [ k, b ] = GetLineEquation(A,B)% line equation
    by 2 points
```

Автор выражает благодарность профессору Вадиму Никитовичу Козлову за полезные замечания и внимание к работе.

**Литература**

1. Козлов В. Н. Элементы математической теории зрительного восприятия. –М., Изд. ЦПИ , 2001., с. 36-52.

2. Кудрявцев В. Б., Гасанов Э. Э., Подколзин А. С. Введение в теорию интеллектуальных систем. – М.: Изд. отд. ф–та ВМиК, МАКС Пресс, 2006, с. 26-32.

# ОБ АЛГЕБРАИЗАЦИИ МОДЕЛИ *K*-ГИПЕРПРОСТРАНСТВА СХ-ГИПЕРТОПОГРАФОВ: ОПЕРАЦИИ ТРАНСФОРМАЦИИ – РАЗВИТИЯ

**Баранович А.Е., Боровиков Д.В., Лакуша Е.Л.**
**(Российский государственный гуманитарный университет, Институт информационных наук и технологий безопасности)**
*barae@rambler.ru, dvborovikov@gmail.com*

В работах [1, 2, 4] исследована универсальная модель информационной составляющей сложных систем, параметрически поглощающая известные классы моделей представления декларативных знаний. В качестве абстрактной экспликации модели определено $k$-гиперпространство СХ-гипертопографов (СХ-$\eta\tau$-графов) [1, 2]. В работе [4] данная модель обобщена до понятия СХ-гипертопосети, моделирующей как декларативные, так и процедурные знания в их сетевой интерпретации.

СХ-$\eta\tau$-граф представляет собой модель статического состояния сложной системы. Для моделирования динамики функционирования системы, наряду с сетевым подходом (нейронные сети, сети Петри и т.п.) в [1] предложено использовать и автоматно-алгебраический подход. Данное направление связано с представлением модели в виде строго формализованной алгебраической системы и опирается на хорошо разработанный механизм алгебраизации множества-носителя (алгебраическая система [6] - множество $G$ («носитель») с заданным на нём набором операций и отношений («сигнатура»), удовлетворяющим некоторой системе аксиом).

В отношении модели классического графа наиболее распространены следующие алгебраические операции (бинарные и унарные) [6]: объединение графов, пересечение графов, удаление вершины, добавление вершины и др. В свою очередь, при рассмотрении операций, связанных с обработкой знаний, наряду с указанными типами операций, возникают и новые, предметно-ориентированные, в частности, операции трансформации/развития, слияния [1], сборки [5] и т.д.

В настоящей работе излагаются результаты алгоритмического синтеза операций трансформации и развития СХ-$\eta\tau$-графов в сигнатуре $\Psi^{\Phi}_{G^{k}_{\eta\tau}x}$, где $\Phi$ есть некоторое, вполне определенное, множество *элементарных трансформаций*. Синтезированные операции положены в основу процедур и программных средств автоматизации

использования декларативно-процедурных знаний [3].

В работе [1] введены понятия *элементарных трансформаций, трансформации, развития* и *глубины трансформации* графа $g$, связанные с вполне определенными алгоритмическими преобразованиями $g$ на множестве $G \equiv \{g\}$. *Элементарные трансформации* порождают на $G$ *бинарные алгебраические операции* типа $\varphi(g, g^*) \equiv \overline{g}$, где $g,\ g^*,\ \overline{g} \in G$. Там же ([1]) доказаны утверждения о *порождении* множеством элементарных трансформаций *Ф сигнатуры операций* $\Psi_G^{\Phi}$ на множестве $G$ и *о существовании* для любых конечномерных графов $g,\ \overline{g} \in G$, по крайней мере, одной операции трансформации $TRF(g) \equiv \bar{g}$, преобразующей исходный (трансформируемый) граф $g$ в граф (результирующий) $\bar{g}$.

Редуцируем задачу алгебраизации классической модели графа на случай СХ-$\eta\tau$-графов.

*Определение 1. Элементарная трансформация* СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$: $(V^{\mathrm{k}}_{\tau}, E^{\mathrm{k}}_{\eta\tau}, \{P^{V^k_{\tau}}, P^{E^k_{\eta\tau}}\})$ на $G^k_{\eta\tau}x$ есть *преобразование* типа: *добавить* к $g^k_{\eta\tau}x$ произвольное *гипертопоребро* $e^k_{\eta\tau} \in \tilde{X}^{(k+1)}$ (обозначим его через $\varphi^+_{e^k_{\eta\tau}}(g^k_{\eta\tau}x)$); *добавить* к $g^k_{\eta\tau}x$ произвольную *гипертоповершину* $v^k_{\tau} \in X^{(k)}$ $(\varphi^+_{v^k_{\tau}}(g^k_{\eta\tau}x))$; *исключить* из $g^k_{\eta\tau}x$ произвольное *гипертопоребро* $e^k_{\eta\tau} \in \tilde{X}^{(k+1)}$ $(\varphi^-_{e^k_{\eta\tau}}(g^k_{\eta\tau}x))$; *исключить* из $g^k_{\eta\tau}x$ произвольную *гипертоповершину* $v^k_{\tau} \in X^{(k)}$ $(\varphi^-_{v^k_{\tau}}(g^k_{\eta\tau}x))$; *добавить цвет* $q \in P$ к подмножеству цветов произвольной *гипертоповершины* $v^k_{\tau}$ СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$ $(\varphi^+_{qv^k_{\tau}}(g^k_{\eta\tau}x))$; *исключить цвет* $q \in P$ из подмножества цветов произвольной *гипертоповершины* $v^k_{\tau}$ СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$ $(\varphi^-_{qv^k_{\tau}}(g^k_{\eta\tau}x))$; *добавить цвет* $q \in P$ к подмножеству цветов произвольного *гипертопоребра* $e^k_{\eta\tau}$ СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$ $(\varphi^+_{qe^k_{\eta\tau}}(g^k_{\eta\tau}x))$; *исключить цвет* $q \in P$ из подмножества цветов произвольного *гипертопоребра* $e^k_{\eta\tau}$ СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$ $(\varphi^-_{qe^k_{\eta\tau}}(g^k_{\eta\tau}x))$; *пустое преобразование* $\varphi_{\emptyset}(g^k_{\eta\tau}x)$ $\triangleleft$

*Элементарные трансформации* порождают на $G^k_{\eta\tau}x$ *бинарные алгебраические операции* типа $\varphi(g^k_{\eta\tau}x, {}^*g^k_{\eta\tau}) \equiv \overline{g^k_{\eta\tau}x}$, где $g^k_{\eta\tau}x$, $\overline{g^k_{\eta\tau}x} \in G^k_{\eta\tau}x$ и ${}^*g^k_{\eta\tau}$ - обобщенное обозначение для вышеуказанных порождающих элементов СХ-$\eta\tau$-графа. Базовым отношением на $G^k_{\eta\tau}x$ при этом являются бинарные отношения *тождества / различия*

элементов $G^k_{\eta\tau}x$.

*Утверждение 1.* Множество элементарных трансформаций $\Phi$ - $\{\varphi^+_{v^k_\tau}(g^k_{\eta\tau}x)$, $\varphi^-_{v^k_\tau}(g^k_{\eta\tau}x)$, $\varphi^+_{e^k_{\eta\tau}}(g^k_{\eta\tau}x)$, $\varphi^-_{e^k_{\eta\tau}}(g^k_{\eta\tau}x)$, $\varphi^+_{qv^k_\tau}(g^k_{\eta\tau}x)$, $\varphi^-_{qv^k_\tau}(g^k_{\eta\tau}x)$, $\varphi^+_{qe^k_{\eta\tau}}(g^k_{\eta\tau}x)$, $\varphi^-_{qe^k_{\eta\tau}}(g^k_{\eta\tau}x)$, $\varphi_\emptyset(g^k_{\eta\tau}x)\}$ порождает *сигнатуру* $\Psi^\Phi_{G^k_{\eta\tau}x}$ *на множестве* $G^k_{\eta\tau}x$ $\triangleleft$

*Определение 2. Трансформация* СХ-$\eta\tau$-графа $TRF(g^k_{\eta\tau}x) \equiv \overline{g^k_{\eta\tau}x}$ на $G^k_{\eta\tau}x$ есть последовательное выполнение некоторой, вполне определенной, совокупности элементарных трансформаций, преобразующих исходный трансформируемый СХ-$\eta\tau$-граф $g^k_{\eta\tau}x$ в результирующий СХ-$\eta\tau$-граф $\overline{g^k_{\eta\tau}x}$ $\triangleleft$

*Определение 3. Развитие* СХ-$\eta\tau$-графа $RV(g^k_{\eta\tau}x) \equiv \overline{g^k_{\eta\tau}x}$ на $G^k_{\eta\tau}x$ есть трансформация исходного СХ-$\eta\tau$-графа $g^k_{\eta\tau}x$ в СХ-$\eta\tau$ граф $\overline{g^k_{\eta\tau}x}$ при условии $g^k_{\eta\tau}x \subseteq \overline{g^k_{\eta\tau}x}$ $\triangleleft$

*Утверждение 2.* Для любых конечномерных СХ-$\eta\tau$-графов $g^k_{\eta\tau}x$ и $\overline{g^k_{\eta\tau}x}$ множества $G^k_{\eta\tau}x$ существует, по крайней мере, одна операция трансформации $TRF(g^k_{\eta\tau}x) \equiv \overline{g^k_{\eta\tau}x}$, преобразующая трансформируемый СХ-$\eta\tau$-граф $g^k_{\eta\tau}x$ в результирующий СХ-$\eta\tau$-граф $\overline{g^k_{\eta\tau}x}$ $\triangleleft$

*Определение 4. Глубина трансформации* $\gamma$ $\{TRF(g^k_{\eta\tau}x) \equiv \overline{g^k_{\eta\tau}x}\}$ *на* $G^k_{\eta\tau}x$ есть число элементарных трансформаций в трансформации $TRF(g^k_{\eta\tau}x) \equiv \overline{g^k_{\eta\tau}x}$ $\triangleleft$

В процессе исследования операции трансформации $TRF$ СХ-$\eta\tau$-графов $g^k_{\eta\tau}x, \overline{g^k_{\eta\tau}x}$, представленных множествами $V^k_\tau, \overline{V^k_\tau}, E^k_{\eta\tau}, \overline{E^k_{\eta\tau}}$ и подмножествами цветов $\{\{p_{v^k_\tau}\}\}$, $\{\{\overline{p_{v^k_\tau}}\}\}$, $\{\{p_{e^k_{\eta\tau}}\}\}$, $\{\{\overline{p_{e^k_{\eta\tau}}}\}\}$, предложена и обоснована теоретико-множественная интерпретация её алгоритмической реализации - ТСХ-процедура.

*Утверждение 3.* ТСХ-процедура на конечных СХ-$\eta\tau$-графах всегда сходится за конечное число шагов $\triangleleft$

*Утверждение 4.* ТСХ-процедура является процедурой трансформации СХ-$\eta\tau$-графов минимальной глубины $\triangleleft$

*Утверждение 5.* Операционная сложность алгоритмической реализации ТСХ-процедуры не превосходит $T(n) = O\,(m \cdot \log s \cdot \log t)$, где $m = \max\{|V^k_\tau| \cdot |\{\{p_{v^k_\tau}\}\}|, |E^k_{\eta\tau}| \cdot |\{\{p_{e^k_{\eta\tau}}\}\}|\}$, $s = \max\{|V^k_\tau|, |E^k_{\eta\tau}|\}$, $t = \max\{|\{\{p_{v^k_\tau}\}\}|, |\overline{\{\{p_{v^k_\tau}\}\}}|, |\{\{p_{e^k_{\eta\tau}}\}\}|, |\overline{\{\{p_{e^k_{\eta\tau}}\}\}}|\}$ $\triangleleft$

Для операции развития $RV$ (RCX - процедура) СХ-$\eta\tau$-графов

получены аналогичные результаты.

В завершении изложения подчеркнем, что *алгебраическая модель* характеризует статическую картину отношений элементов моделируемой системы. Моделью же *преобразования* статических состояний системы является *операция* [4]. Объединение моделей статического состояния систем (*алгебраическая модель*) и модели изменений состояний модели в текущем настоящем (*алгебра*) порождает, в итоге, динамическую модель *алгебраической системы*. Введение операций на СХ-$\eta\tau$-графах влечет порождение вполне определенной *метаалгебры* («алгебры моделей») или в иной, вышеупомянутой постановке, *метаалгебраической* системы (с вполне определенным множеством-носителем $V$).

**Литература**

1. Баранович А.Е. Основы алгебраизации модели: алгоритмическая концепция: в кн. «Семиотико-хроматические гипертопографы. Введение в аксиоматическую теорию: информационный аспект (прил. 1)». – М.: МО РФ, 2003. – 404 с.

2. Баранович А.Е. $K$-гиперпространство семиотико-хроматических гипертопографов как универсальная модель представления фактографических знаний // Мат. IX Междунар. конф. «Интеллект. сист. и компьют. науки». Т. 1. Ч. 1. – М., МГУ, 2006. – с. 53-55.

3. Баранович А.Е., Баранович А.А., Лишин Н.А. Исчисление ценности прагматической информации в интеллектуальной программной среде «АКСИОН» // Тр. XI национ. конф. по искусственному интеллекту с междунар. участ. Т.3. – М., ЛЕНАНД, 2008. – с. 364-372.

4. Баранович А.Е. Семиотико-хроматические гипертопосети: унифицированная модель представления знаний // Открытые семантические технологии проектирования интеллектуальных систем: материалы Междунар. научн.-техн. конф. – Минск, БГУИР, 2011. – с. 71-86.

5. Зайцев Д.В. О сложности сборки и вложения графов // Диссер. на соиск. учен. степ. канд. физ.-мат. наук. – М., МГУ, мех.-мат. фак-т, 2007. – 106 с.

6. Капитонова Ю.В. Летичевский А.А., Луцкий Г.М. Лекции по дискретной математике. – СПб.: БХВ-Петербург, 2004. – 624 с.

7. Яблонский С.В. Введение в дискретную математику: Учебное пособие для вузов // Под ред. В.А. Садовничего. – 3-е изд., стер. – М.: Высш. шк., 2002. – 384 с.

## О НЕКОТОРЫХ РЕЗУЛЬТАТАХ СРАВНИТЕЛЬНОГО АНАЛИЗА МОДЕЛЕЙ МУЗЫКАЛЬНОГО И ВЕРБАЛЬНОГО ТЕКСТОВ

**Баранович А.Е., Иглицкая С.М. (Российский государственный гуманитарный университет, Институт информационных наук и технологий безопасности)**

*barae@rambler.ru, sofa.sofa@mail.ru, www.samtcenter.ru*

Обширный спектр проблем, связанных с системным анализом и математическим моделированием музыкального текста (МТ), представляет собой весьма мало изученную область. Обзор доступных источников показывает, что существующие методология и инструментарий исследований не позволяют решить целый ряд задач, связанных с характеризацией естественнонаучного базиса МТ, определяемого его принадлежностью к сфере информационной коммуникации вполне определенного подкласса класса антропоморфных интеллектуальных систем (ИС) [6].

Один из возможных подходов к исследованию МТ основывается на использовании (по аналогии) известных (вполне определённым образом модифицированных), методов структурно-алгебраического и семантико-прагматического анализа вербального текста (ВТ). В настоящей работе отражены некоторые результаты исследования МТ так называемого строгого стиля, представленного моделью дискретных сообщений Дж. фон Неймана нулевого и первого приближений, в задаче оценки пропускной способности дискретного канала связи по К.Шеннону. Последующие исследования предполагают введение в разрабатываемый аппарат моделирования МТ информационных моделей семантико-прагматического типа, ориентированных на исследование особенностей семантической музыкальной коммуникации коллектива ИС.

Выдвигалась следующая конструктивная гипотеза: $C_m > C_v$, где $C_v$ и $C_m$ - пропускные способности каналов соответственно вербальной и музыкальной коммуникации, пропорциональные энтропии источника дискретных сообщений. Для вычисления последней используется формула Шеннона: $H(A) = -\sum_{i=1}^{m} P(a_i) \log P(a_i)$; для случая коррелированных символов (условной энтропии): $H(A'/A) = -\sum_{i=1}^{m} P(a_i) \sum_{j=1}^{m} P(a'_j/a_i) \log P(a'_j/a_i)$, где $a_i$, $a_j$ - символы алфавита $A$ источника сообщений, $|A| = m$, $P(a_i)$ - вероятность появления символа $a_i$, $P(a'_j/a_i)$ - условная вероятность появления символа $a_j$

после символа $a_i$ [5].

Для представления МТ в виде конечной совокупности конечных последовательностей символов, на основе существующего музыкального направления так называемого строгого стиля (С.С.) (историческое и художественно-стилистическое понятие, относящееся к хоровой полифонической музыке эпохи Ренессанса [7, 8]) был составлен определенный «музыкальный алфавит», генерируемые на базе которого мелодии могут служить моделью композиторских мелодий данного стиля разной степени приближения.

Последовательная семиотическая модель Дж. фон Неймана [1] представляет собой конечные (длины $l$) последовательности символов алфавита $A$:
$A : \sigma_i = \sigma_{i_1}, ..., \sigma_{i_l}, l \leq L \leq \infty, \sigma_{i_j} \in A, |A| = z, z \leq Z \leq \infty$ для $\forall i, j$, $j = \overline{1, l}$, $i = \overline{1, N}$, $N \leq \sum_{k=1}^{L} Z^k$

В *модели нулевого приближения* не вводится никаких ограничений на порядок следования символов алфавита; в *модели первого приближения* имеются ограничения на пары подряд идущих символов (запретные биграммы), определяемые бинарной матрицей $B = (\{b_{ij}\}), i, j = \overline{1, Z}$, где $b_{ij} = 1$ для разрешенной, и $b_{ij} = 0$ - для запрещенной пары символов алфавита $A$.

Количество слов длины, не превышающей $l$, выражается следующим образом:

$S_0(l) = \sum_{k=1}^{l} Z^k$ в модели нулевого приближения,

$S_1(l) = \sum_{k=2}^{l} \left( \sum_{i,j=1}^{Z} b_{ij}^{(k-1)} \right) + Z$ в модели первого приближения.

Получены следующие числовые характеристики для энтропии МТ ($H_m$) и ВТ ($H_v$):

в модели нулевого приближения $H_v \approx 5,04$, $H_m \approx 7,51$;

в модели первого приближения $H_v \approx 4.6398$, $H_m \approx 4.6018$.

Анализ полученных результатов позволяет сформулировать следующие промежуточные выводы:

1. Алфавит МТ обладает по сравнению с ВТ гораздо большей мощностью, однако МТ в силу своей художественной природы обладает и значительно более строгим набором запретов на сочетания символов.
2. Пропускная способность канала коммуникации зависит от сочетания данных параметров - мощности алфавита и жесткости правил.
3. Поскольку результаты вычислений, проведенные на модели МТ

С.С., представляют собой оценку снизу всех аналогичных результатов для подавляющего большинства музыкальных текстов (МТ С.С. обладает минимальным алфавитом и подчиняется максимально строгим правилам, регламентирующим запретные последовательности нот), то уже для одноголосия гипотеза о большей пропускной способности канала музыкальной коммуникации представляется вполне обоснованной. Для двухголосия же примерная оценка в модели нулевого приближения: $H_m = \log_2 6560 \approx 16,01$.

При рассмотрении вопросов построения модели семантики МТ необходимо различать понятия объективной семантики, присущей самой информации, и субъективной (прагматической) семантики, зависящей от воспринимающего субъекта: « $<\ldots>$ *обвективная семантика* информации характеризует информационные формы существования материальных систем объективной реальности и взаимосвязана с формой, структурой и организацией материальных систем; $<\ldots>$ *семантика субъективная* (прагматическая) интерпретируется как динамический информационный образ объективной семантики, инициализированный в подсистеме знаний воспринимающей интеллектуальной системы» [3].

Для исследования объективной семантики МТ предлагается использовать известную модель $k$-гиперпространства СХ-гипертопогра-фов как универсальную абстрактную модель информационной составляющей сложных систем([1, 2]). Моделирование одномерных и многомерных линейных структур (ВТ, одно- и многоголосного МТ, графики) предполагает использование модифицированной модели СХ-гипертопографа, с расширенным множеством-носителем, характеризующим множественность различимых (посредством хроматических атрибутов) экземпляров односортных элементов. Дальнейшая топологизация множества-носителя реализуется согласно классической модели СХ-гипертопографа.

Для моделирования МТ как динамического процесса возможно использование модели СХ-гипертопосетей [4], где в качестве статических состояний системы, представленных СХ-гипертопографами, рассматриваются всевозможные вертикальные созвучия, априорно обладающие собственной сложной структурой отношений (особенно для полифонической музыки).

Приоритет в выборе того или иного подхода зависит от стилистических особенностей анализируемого (моделируемого) МТ: если

обладающую развитой совокупностью структурных связей сонатную форму целесообразно представить в виде статической модели, то аморфный по форме (то есть характеризуемый слаборазвитыми связями между удаленными элементами), но обладающий строго детерминированными правилами соотношения соседних созвучий МТ С.С. логичным представляется исследовать с применением аппарата моделирования динамических систем.

**Литература**

1. Баранович А.Е. Структурное метамоделирование телеологических информационных процессов в интеллектуальных системах. - М., МО РФ, 2002. - 316 с.

2. Баранович А.Е. Семиотико-хроматические гипертопографы. Введение в аксиоматическую теорию: информационный аспект. - М.: МО РФ, 2003. - 404 с.

3. Баранович А.Е. Семантические аспекты информационной безопасности: концентрация знаний / Вестник РГГУ. Сер. «Информатика. Защита информации. Математика». - М., РГГУ, 2011 (в печ.).

4. Баранович А.Е. Семиотико-хроматические гипертопосети: унифицированная модель представления знаний / Открытые семант. технол. проектир. интеллект. систем (OSTIS-2011): мат. Междунар. научн.-техн. конф. - Минск, БГУИР, 2011. - с. 71-86.

5. Гуров И.П. Основы теории информации и передачи сигналов: электрон. учеб. по дисципл. «Теория информации и передачи сигналов». - СПб., СПбГУ-ИТМО, Кафедра компьютерных технологий. http://de.ifmo.ru/bk_netra/page.php?tutindex=11. Дата доступа: 20.04.2011.

6. Иглицкая С.М. К вопросу структурно-алгебраического и семантико-прагматического анализа музыкального текста / Вестник РГГУ. Сер. «Информатика. Защита информации. Математика». - М., РГГУ, 2011 (в печ.).

7. Музыкальная энциклопедия в 6 томах, гл. ред. Ю.В. Келдыш. Т. 5. Симон - Хейлер. - М.: Советская энциклопедия, 1981. - 1056 с., ил.

8. Фраёнов В.П. Учебник полифонии. - М.: Музыка, 1987. - 207 с., нот.

# МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ДЛЯ ОПТИМИЗАЦИИ ИНДИВИДУАЛЬНОЙ ПРАКТИЧЕСКОЙ РАБОТЫ УЧАЩЕГОСЯ

**Биштейнов Д.А. (Смоленский государственный университет)**
*bisteinoff@gmail.com*

*Индивидуализация обучения – актуальная задача в нашей стране. В этой статье мы приводим модель представления теоретического материала учебника и практических заданий задачника с целью оптимизации управления индивидуальной практической работой ученика в классе за счет автоматизации выполнения некоторых задач учителя.*

*Ключевые слова: индивидуализация образовательного процесса, модель, математическое моделирование, графовая модель, система автоматизированного проектирования работы учителя (САПР учителя).*

В настоящее время необходимость индивидуализации обучения осознается не только учеными-новаторами, но и государством. В Концепции долгосрочного социально-экономического развития Российской Федерации до 2020 г. указывается: «*Развитие системы общего образования предусматривает* ***индивидуализацию****, ориентацию на практические навыки и фундаментальные умения. . .* ». В современных условиях основным источником образовательного запроса к системе образования становится личность учащегося. Его интересы, потребности, способности, мотивы должны во все большей степени учитываться при проектировании и организации процесса обучения [5]. Интерес представляет изучение не только учебно-воспитательного процесса, но и процесса индивидуального развития личности, в рамках сохраняющейся классно-урочной системы. Для третьей ступени обучения существует нормативно-правовая база для обучения по индивидуальным учебным планам, в профильных классах. Однако по-прежнему не дается ответ на вопрос, как обучать в группе, чтобы не сдерживать индивидуальное развитие учащегося. Поставим **задачу** *оптимизации работы учителя, направленной на создание условий, в которых учащийся может выполнять практические задания на уроке в индивидуальном темпе.*

Актуальность реализации планирования учебного процесса в системах автоатизированного проектирования работы (САПР) учителя была рассмотрена в [1]. Представляет интерес САПР учите-

ля «Траектория обучения», которая рассчитывает индивидуальную траекторию обучения класса (группы) или отдельного учащегося [5, с. 71-78].

Предложим математическую модель учебника и задачника, которая может быть положена в основу САПР учителя, позволяющей решить поставленную нами задачу.

Имеется модель представления учебного материала учебника T в виде последовательности $\{L_i\}$, i = 1...n, где $L_i$ – это модель урока, которая представляет собой массив $L_i(C(L_i), Y(L_i), M(L_i), K(L_i))$, где $C(L_i)$ – множество целей урока $L_i$ из множества целей C, $Y(L_i)$ – тип урока $L_i$ из множества типов Y, $M(L_i)$ – множество требуемых материалов для проведения урока $L_i$ из множества материалов M, $K(L_i)$ – граф, моделирующий теоретический материал урока $L_i$, который мы будем называть *знаниевым графом*. Знаниевый граф с математической точки зрения является графовой моделью цели обучения [5, с. 64-68]. Однако, на наш взгляд, неверно называть его целью обучения, так как он может содержать вершины, ассоциированные с элементами знаний, которые не будут являться обязательными для изучения или вовсе не будут входить в курс изучения предмета в рамках данного УМК.

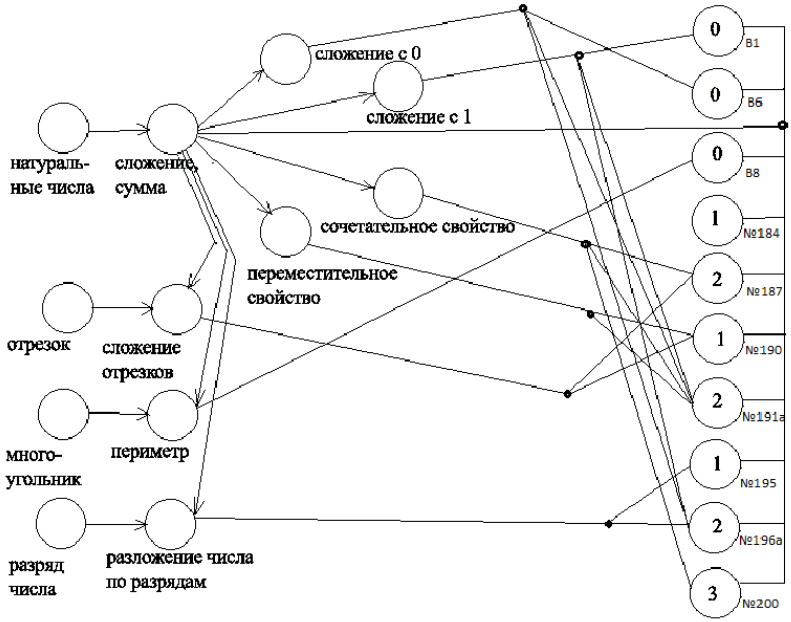
Знаниевый граф является орграфом [2]. Его вершины ассоциируются с элементами знания по некоторой теме, дуги – с наличием логических и причинно-следственных связей между элементарными компонентами содержания обучения [4, с. 228]. Под теоретическим материалом урока мы будем понимать все элементарные компоненты, которые необходимо узнать учащимся по плану урока и на которые опирается изложение нового материала.

Пусть граф P – это граф со взвешенными вершинами, который мы обобщенно назовем *графом задач* (кроме задач, могут быть упражнения, вопросы, задания и др., а также к задачам иногда может быть целесообразно отнести дополнительный теоретический материал повышенной сложности). Вес вершин графа P означает *сложность* задания. Смоделируем граф G ( $V(K) \cup V(P)$, $E(V(K) \cup V(P))$ ), где V(K) – множество вершин знаниевого графа K, V(P) – множество вершин графа задач P, $E(V(K) \cup V(P))$ – множество ребер, содержащее все ребра графа K и все ребра между вершинами графа K и вершинами графа P, которые устанавливаются на основании взаимного соответствия элементов теоретического

материала из K задачам из P, для решения которых они необходимы (рис. 1). Отметим, что граф G содержит подграф $G_1 = K(L_i)$ и $G_2$( V(G), E(G) - E(K) ). Подграф $G_1$ ориентированный. Подграф $G_2$ неориентированный, двудольный.

Отметим, что для ряда УМК может быть целесообразнее построить модель параграфа, темы, раздела. Мы также предполагаем, что сложность задания может быть прямо-пропорциональной степени исхода вершины, ассоциированной с ним.

Приведем пример модели параграфа «Сложение натуральных чисел и его свойства» из учебника 5 класса Н.Я. Виленкина [3]. Задачник содержит более 100 задач для отработки материала этой темы. Кроме того, число задач может быть увеличено за счет других задачников. Для нашей модели мы выбрали 10 задач таким образом, чтобы все вершины графа задач были смежны хотя бы одной вершине знаниевого графа.




Итак, в данной статье мы предложили теоретическую модель

учебника и задачника для системы автоматизированного проектирования работы учителя, потенциально расширяющую функции САПР учителя «Траектория обучения» и позволяющую оптимизировать трудозатраты учителя на организацию индивидуальной практической работы учащихся в группе.

**Литература**

1. Биштейнов Д. А. Актуальность реализации планирования учебного процесса в системах автоатизированного проектирования работы учителя / Д.А. Биштейнов // Методология и методика информатизации образования в многоступенчатой структуре высшей школы: материалы Всероссийской научно-практической конференции (7-8 декабря 2009 года). – Смоленск: Изд-во СмолГУ, 2009. – С. 4–6.

2. Кристофидес Н. Теория графов. Алгоритмический подход / Н. Кристофидес. – М.: Мир, 1978. – 432 с.

3. Математика: Учеб. для 5 кл. общеобразоват. учреждений / Н.Я. Виленкин, В.И. Жохов, А.С. Чесноков, С.И. Шварцбурд. – 17-е изд., перераб. – М.: Мнемозина, 2005. – С. 33-41.

4. Новиков А. М. Развитие отечественного образования. Полемические размышления / А.М. Новиков – М.: Изд-во «Эгвес», 2005. – 256 с.

5. Сенькина Г. Е. Методы математического моделирования в обучении / Г.Е. Сенькина, Е.П. Емельченков, О.М. Киселева. – Смоленск: Смол. гос. ун-т, 2007. – 112 с.

# ЭВОЛЮЦИОННЫЕ СИМВОЛЬНЫЕ ВЫЧИСЛЕНИЯ: МЕТОДИКА И ИНСТРУМЕНТАРИЙ

**Борченинов Я.В., Окуловский Ю.С.**
**(Уральский государственный университет им. А.М. Горького)**
*yuri.okulovsky@gmail.com*

Эволюционные символьные методы (ЭСВ) широко применяются для аппроксимации данных, поиска инвариантов и физических законов, классификации, решении дифференциальных уравнений и других задач. ЭСВ используют генетические алгоритмы для направленного перебора синтаксических конструкций. По конструкции строится выражение, которое оценивается на соответствие исходным данным. Первоначально этот метод был предложен, по всей видимости, Джоном Коза (см., например, [1]) для автоматической генерации программ. Метод был далее адаптирован для других задач: для построения выражений [2], для решения дифференциальных уравнений [3], для восстановления естественных законов из экспериментальных данных [4] и т.д.

Настоящая работа посвящена проработке следующих проблем ЭСВ. Для существующих решений характерна специализация для работы с числами с плавающей точкой. Другие домены (целые числа, нечеткие логические переменные, функции) используются редко, практически не встречается совместное использование доменов (например, логических выражений и чисел), не решены возникающие проблемы с ошибками типизации. Также в исследованиях ЭСВ непропорциональное внимание уделяется замысловатым способам кодирования и решению проблем, которое это кодирование вызывает, хотя, как мы покажем далее, эта проблема решается одним простым и элегантным обобщением. Кроме того, игнорируется опыт систем компьютерной алгебры и дедуктивных преобразований выражений. В лучшем случае, полученное в ходе ЭСВ выражение упрощается на выходе применением какой-либо системы компьютерной алгебры. Нам представляется, что полноценная интеграция дедуктивных преобразований оправдана как с математической, так и с философской точек зрения. Наконец, мало проработаны метрики оценки выражений, что приводит к появлению сложных, избыточных и неэстетических выражений.

В настоящей работе мы изложим методику проведения ЭСВ, решающую указанные проблемы. Данная методика воплощена в биб-

лиотеке ЭСВ на языке C#, которая в данный момент находится в состоянии прототипа. В дальнейшем эта библиотека будет опубликована под лицензией GPL v.3.

Методика использует следующее обобщение генетических алгоритмов. Исторически, при работе со сложными структурами данных, генетический алгоритм кодирует их в виде битовых массивов или строк. Мутация сводится к отрицанию бита или изменению буквы, а скрещивание – к обмену фрагментами массивов. Этот подход приводит к разрушению целостности структуры данных, и вынуждает изобретать методы для ее поддержания. Мы используем обобщенную формулировку генетического алгоритма, которая повторяет классическую, но разрешает обработку произвольных структур данных, для которых определены нульарный оператор генерации, унарный оператор мутации и бинарный оператор скрещивания. Обобщенный генетический алгоритм позволяет выбрать интуитивное, простое кодирование для задачи.

Кодирование выражений выполняется в виде дерева. Узлами дерева являются операции вида $f : D_{i_1} \times \ldots \times D_{i_k} \to D_j$, где $D_s$ – произвольные домены (типы) данных. Листьями дерева являются типизированные константы и переменные. Каждый узел дерева содержит тип данных, возвращаемых выражением, закодированном в соответствующем поддереве.

Операция генерации начального гена для выражения типа $D$ состоит в создании дерева из единственного листа – константы соответствующего типа.

Операции мутации определяются на основе правил. Данный подход заимствован из систем компьютерной алгебры, в которых все свойства операций определяются списками преобразований (таких как $(x+y)z \Rightarrow xz+yx$). В системах компьютерной алгебры эти операции дедуктивны, они изменяют лишь форму выражения, но не кодируемую им функцию. В ЭСВ, мы используем также индуктивные правила, изменяющую как структуру, так и кодируемую функцию. Все правила сохраняют тип поддерева, чем обеспечивается корректность собираемого выражения.

Приведем примеры индуктивных правил. Правила замены константы вида $c_{\text{bool}} \Rightarrow \neg c$ и $c_{\text{real}} \Rightarrow c(1 + \alpha/2 - \text{rnd}(\alpha))$ позволяют «подгонять» значение константы $c$. Правила вида $N_{\text{T}} \Rightarrow x_{\text{T}}$ позволяют заменить поддерево $N_{\text{T}}$ типа T на переменную того же типа.

Операции вводятся правилами ввода: $N_{\text{int}} \Rightarrow N + 0$, $N_{\text{int}} \Rightarrow N \cdot 1$, $N_{\text{int}} \Rightarrow \text{TRUE}?N : 0$, $N_{\text{bool}} \Rightarrow N \wedge \text{TRUE}$, $N_{\text{bool}} \Rightarrow c_{\text{int}} < d_{\text{int}}$ и т.д. Операция также может быть выведена универсальным способом $[f(N_{\text{T}}, \ldots)]_{\text{T}} \Rightarrow N_{\text{T}}$. Приведем примеры дедуктивных правил. Структурные правила используют ассоциативность и коммутативность некоторых операций, и перестраивают дерево в некую каноническую структуру, а также сортируют аргументы коммутативных функций так, чтобы константы оказывались в конце дерева. Правила сокращения выполняют такие замены, как $2 + 2 \Rightarrow 4$ и $0 \cdot x \Rightarrow 0$. Также введены правила преобразований вида $N^2_{\text{real}} \Leftrightarrow N \cdot N$. Применение в левую сторону делает выражение более эстетически привлекательным, в правую – облегчает применение индуктивных правил, делая, например, возможной двухходовую замену $x \cdot x \Rightarrow x^2 \Rightarrow x^{1.9}$. Для поддержки правил в программном коде была реализована система условий и замен фрагментов дерева, по идеологии схожая с применением регулярных выражений к строкам.

Операция скрещивания в простейшем случае сводится к обмену поддеревьями одного типа. Кроме того, возможно сложное скрещивание с введением условного перехода $A, B \Rightarrow TRUE?A : B$, операций полусуммы $A_{\text{real}}, B_{\text{real}} \Rightarrow (A + B)/2$ или геометрического среднего $A_{\text{real}}, B_{\text{real}} \Rightarrow \sqrt{AB}$.

После построения выражения, его необходимо оценить. Базовой метрикой оценки является соответствие, т.е. то, насколько точно выражение описывает данные. Для оптимизации вычисления этой оценки, используются инструменты языка C#: лямбда-выражения и Expression Trees [5]. Эти инструменты позволяют скомпилировать для дерева операций лямбда-выражение, вычисление которого значительно быстрее обхода дерева. Лямбда выражение задается для каждого узла, но компилируется только для корня. Для константы строится выражение $X \mapsto c$ ($X$ – кортеж всех аргументов построенного выражения), для переменной – $X \mapsto X[i]$, для операции $f$ – $X \mapsto f(g_1(X), \ldots, g_k(X)$, где $g_i$ – лямбда-выражения для детей $f$.

Помимо соответствия, используются и другие метрики. Метрика сложности определяет вычислительную сложность выражения и стремится минимизировать ее, метрика размера приводит к усечению неиспользуемых поддеревьев, метрика эстетичности выбирает наиболее эстетически привлекательные выражения. Данное направление в настоящий момент является предметом дополнительных ис-

следований.

Приведенная методика была протестирована на обычных для ЭСВ задачах аппроксимации, поиска инвариантов и классификации. Создан тестовый набор «стандартных» задач, на котором мы планируем сравнивать различные реализации ЭСВ в дальнейшем.

В заключение кратко сформулируем основные преимущества нашей методики. За счет разрешения использования разных доменов данных, методика обобщает все примеры ЭСВ, известные ранее. Кодирование выражений происходит простым и интуитивным путем. Введение новых операций производится с помощью правил, что позволяет расширять ЭСВ произвольными операциями и доменами. Помимо индуктивных, используются дедуктивные правила. Введены различные метрики оценки решения, что позволяет получить не только верные, но также неизбыточные и эстетически привлекательные решения.

Разработанная методика и инструментарий позволяют проводить дальнейшие исследования ЭСВ в следующих областях: манипулирование частотой применения правил разных типов для получения оптимальных результатов; исследования оптимального веса метрик для получения решений, сбалансированных по разным параметрам; применение ЭСВ к новым областям, в первую очередь к логике и нечеткой математике.

**Литература**

1. Koza J. Genetic Programming: On the Programming of Computers by Means of Natural Selection // Mit Press, 1992.

2. Ferreira C. Gene Expression Programming:Mathematical Modeling by an Artificial Intelligence // Springer, 2006.

3. Diver D.A. Applications of genetic algorithms to the solution of ordinary differential equations // In *Journal of Physics A: Mathematical and General*, N26, 1993.

4. Schmidt M., Lipson, H. Distilling Free-Form Natural Laws from Experimental Data // In *Science*, Vol. 324, no. 5923, 2009.

5. Troelsen A. Pro C# 2010 and the .NET 4 Platform // Apress, 2010.

# ПРИБЛИЖЕННОЕ РЕШЕНИЕ ЗАДАЧИ О БЛИЗОСТИ В ЕВКЛИДОВОЙ МЕТРИКЕ

**Гасанов Э.Э., Остроухова Е.Н.**
**(Московский государственный университет)**
*el_gasanov@mail.ru*

Мы будем использовать терминологию и обозначения из работы [1].

Пусть $X$ — множество запросов с заданным на нем вероятностным пространством $\langle X, \sigma, \mathbf{P}\rangle$, где $\sigma$ — алгебра подмножеств множества $X$, $\mathbf{P}$ — вероятностная мера на $\sigma$; $Y$ — множество записей (объектов поиска); $\rho$ — бинарное отношение на $X \times Y$, называемое *отношением поиска*. Пятерку $S = \langle X, Y, \rho, \sigma, \mathbf{P}\rangle$ назовем *типом*. Тройку $I = \langle X, V, \rho\rangle$, где $V$ — некоторое конечное подмножество множества $Y$, в дальнейшем называемое *библиотекой*, будем называть *задачей информационного поиска (ЗИП)*, и будем считать, что ЗИП $I = \langle X, V, \rho\rangle$ содержательно состоит в перечислении для произвольно взятого запроса $x \in X$ всех тех и только тех записей $y \in V$ таких, что $x\rho y$.

Пусть $f$ — одноместный предикат, определенный на $X$, то есть $f : X \to \{0, 1\}$. Множество $N_f = \{x \in X : f(x) = 1\}$ назовем *характеристическим множеством предиката* $f$.

Множество $O(y, \rho) = \{x \in X : x\rho y\}$ назовем *тенью записи* $y \in Y$.

Функцию $\chi_{y,\rho} : X \to \{0, 1\}$ такую, что $N_{\chi_{y,\rho}} = O(y, \rho)$ назовем *характеристической функцией записи* $y$.

Пусть $F$ — множество символов одноместных предикатов, определенных на множестве $X$, $G$ — множество символов одноместных переключателей, определенных на множестве $X$. Под переключателем будем понимать функцию, областью значений которой является начальный отрезок натурального ряда. Пару $\mathcal{F} = \langle F, G\rangle$ назовем *базовым множеством*.

Понятие *информационного графа (ИГ) над базовым множеством* $\mathcal{F} = \langle F, G\rangle$ определяется следующим образом. Рассмотрим конечную многополюсную ориентированную сеть. В ней выбираем полюс, который называем корнем. Остальные полюса называются листьями и им приписываются записи из $Y$, причем разным листьям могут быть приписаны одинаковые записи. Некоторые вершины сети называются переключательными и им приписаны переключатели из

множества $G$. Ребра, исходящие из этих вершин, нумеруются подряд, начиная с единицы, и называются переключательными ребрами. Остальные ребра сети называем предикатными и приписываем им предикаты из множества $F$. Таким образом нагруженную многополюсную сеть называем информационным графом над базовым множеством $\mathcal{F} = \langle F, G \rangle$.

*Функционирование информационного графа* определяется следующим образом. Скажем, что предикатное ребро проводит запрос $x \in X$, если $f(x) = 1$. Переключательное ребро, которому приписан номер $n$, проводит запрос $x$, если переключатель, приписанный началу этого ребра, на запросе $x$ принимает значение $n$. Ориентированная цепь ребер проводит запрос $x$, если каждое ребро этой цепи проводит $x$. Запрос $x \in X$ проходит в вершину $\beta$ ИГ, если существует ориентированная цепочка ребер, ведущая из корня в вершину $\beta$ и проводящая запрос $x$. Запись $y$, приписанная листу $\alpha$, попадает в ответ ИГ на запрос $x \in X$, если запрос $x$ проходит в лист $\alpha$. Ответом ИГ $U$ на запрос $x$ назовем множество записей, попавших в ответ ИГ на запрос $x$, и обозначим его $\mathcal{J}_U(x)$. Функцию $\mathcal{J}_U(x) : X \to 2^Y$ будем считать результатом функционирования ИГ $U$.

Для ЗИП $I = \langle X, V, \rho \rangle$ обозначим $\mathcal{J}_I(x) = \{y \in V : x \rho y\}$ — ответ задачи $I$ на запрос $x$, $R(I) = \mathbf{M}_x |\mathcal{J}_I(x)| = \sum\limits_{y \in V} \mathbf{P}(O(y, \rho))$ — средняя длина ответа, или среднее время перечисления ответа.

Пусть нам дана ЗИП $I = \langle X, V, \rho \rangle$. Скажем, что ИГ $U$ решает ЗИП $I = \langle X, V, \rho \rangle$, если $\mathcal{J}_U(x) = \{y \in V : x \rho y\}$. ИГ $U$, решающий ЗИП $I$, будем также называть *допустимым* для $I$. Множество ИГ над базовым множеством $\mathcal{F}$, решающих ЗИП $I$, будем обозначать $\mathcal{U}(I, \mathcal{F})$.

Пусть $\beta$ — некоторая вершина ИГ. Предикат, определенный на множестве запросов, который принимает значение 1 на запросе $x$, если запрос проходит в вершину $\beta$, и 0 — в противном случае, назовем *функцией фильтра* вершины $\beta$ и обозначим $\varphi_\beta(x)$.

*Сложностью ИГ $U$ на запросе $x \in X$* назовем число

$$T(U, x) = \sum_{\beta \in \mathcal{P}} \varphi_\beta(x) + \sum_{\beta \in \mathcal{R} \setminus \mathcal{P}} \psi_\beta \cdot \varphi_\beta(x),$$

где $\psi_\beta$ — количество ребер, исходящих из вершины $\beta$, $\mathcal{P}(U)$ — множество вершин переключения ИГ $U$, $\mathcal{R}(U)$ — множество всех вершин

ИГ $U$.

Скажем, что базовое множество $\mathcal{F}$ — *измеримое*, если каждая функция из $\mathcal{F}$ измерима (относительно $\sigma$). Далее везде будем предполагать, что базовое множество измеримо. В этом случае для любого ИГ $U$ над $\mathcal{F}$ функция $T(U,x)$ как функция от $x$ измерима.

*Сложностью ИГ* $U$ назовем математическое ожидание величины $T(U,x)$, то есть число $T(U) = \mathbf{M}_x T(U,x)$.

*Сложностью в худшем случае* назовем величину $\widehat{T}(U) = \max\limits_{x\in X} T(U,x)$.

Будем рассматривать следующие сложностные характеристики:

$$T'(U,x) = T(U,x) - |\mathcal{J}_I(x)|,\ \ T'(U) = \mathbf{M}_x T'(U,x),\ \ \widehat{T}'(U) = \max_{x\in X} T'(U,x).$$

$T'(U,x)$ характеризует время поиска без перечисления ответа для запроса $x$.

*Объемом* $Q(U)$ *ИГ* $U$ назовем число ребер в графе $U$.

Пусть $I = \langle X, V, \rho\rangle$ — исходная задача информационного поиска. Рассмотрим вспомогательную задачу $\widetilde{I} = \langle X, V, \widetilde{\rho}\rangle$.

Назовем *шумом пары* $\langle I, \widetilde{I}\rangle$ *на запросе* $x$ множество $N(x) = \mathcal{J}_{\widetilde{I}}(x) \setminus \mathcal{J}_I(x)$. Назовем *дефицитом пары* $\langle I, \widetilde{I}\rangle$ *на запросе* $x$ множество $D(x) = \mathcal{J}_I(x) \setminus \mathcal{J}_{\widetilde{I}}(x)$. *Средним шумом пары* $\langle I, \widetilde{I}\rangle$ назовем величину $n(I, \widetilde{I}) = \mathbf{M}_x|N(x)| = \sum\limits_{y\in V} \mathbf{P}(O(y,\widetilde{\rho}) \setminus O(y,\rho))$. *Средним дефицитом пары* $\langle I, \widetilde{I}\rangle$ назовем величину $d(I, \widetilde{I}) = \mathbf{M}_x|D(x)| = \sum_{y\in V} \mathbf{P}(O(y,\rho) \setminus O(y,\widetilde{\rho}))$.

Величину $m(\alpha) = \frac{\alpha n + (1-\alpha)d}{R(I)}$, зависящую от действительного параметра $\alpha \in [0,1]$, называем *ошибкой пары* $\langle I, \widetilde{I}\rangle$ *при цене ошибки* $\alpha$. Величину $\widehat{m} = \sup\limits_{\alpha\in[0,1]} m(\alpha)$ называем *максимальной ошибкой пары* $\langle I, \widetilde{I}\rangle$.

Опишем тип задач поиска, который соответствует двумерной задаче о близости в евклидовой метрике.

Пусть $X = Y = [0,1]^2$ — множества запросов и записей соответственно. Пусть на множестве $X$ задано вероятностное пространство $\langle X, \sigma, \mathbf{P}\rangle$, $\mathbf{P}$ — равномерное распределение на $X$. Отношение поиска $\rho_R$ определено на $X \times Y$ и задается следующим соотношением: $(x_1,x_2)\rho_R(y_1,y_2) \iff \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2} \le R$, где $R \in [0,1]$.

Тогда тип $\langle X, Y, \rho_R, \sigma, \mathbf{P}\rangle$ назовем *типом двумерной задачи о близости в евклидовой метрике.*

Если $a \in \mathbb{R}$, то $]a[$ — наименьшее целое, не меньшее, чем $a$.

Если $\alpha = (\alpha_1, \alpha_2), x = (x_1, x_2) \in \mathbb{R}^2$, то $\alpha x = \alpha_1 x_1 + \alpha_2 x_2$.

Если $\alpha \in \mathbb{R}^2$, $a, b, d \in \mathbb{R}$, $m \in \mathbb{N}$, то обозначим

$$f_{\alpha,d}(x) = \begin{cases} 1, & \text{если } \alpha x \geq d, \\ 0, & \text{если } \alpha x < d, \end{cases} \tag{1}$$

$$F_1 = \{f_{\alpha,d}(x) : \alpha \in \mathbb{R}^2, d \in \mathbb{R}\}, \tag{2}$$

$$g_{\alpha,d}(x) = \begin{cases} 1, & \text{если } \alpha x \leq d, \\ 2, & \text{если } \alpha x > d, \end{cases} \tag{3}$$

$$G_1 = \{g_{\alpha,d}(x) : \alpha \in \mathbb{R}^2, d \in \mathbb{R}\}, \tag{4}$$

$$g_{\alpha,a,b,m}(x) = \left]\frac{\alpha x - a}{b - a} m\right[, \tag{5}$$

$$G_2 = \{g_{\alpha,a,b,m}(x) : \alpha \in \mathbb{R}^2, a, b \in \mathbb{R}, m \in \mathbb{N}\}, \tag{6}$$

$$\mathcal{F} = \langle F_1 \cup \{1\}, G_1 \cup G_2\rangle. \tag{7}$$

Здесь 1 — предикат тождественная единица.

Будем писать $A(n) \preceq B(n)$, если существует константа $c > 0$, что $A(n) \leq cB(n)$, начиная с некоторого номера $n_0$.

**Теорема.** *Пусть $I = \langle X, V, \rho_R\rangle$ — двумерная задача о близости в евклидовой метрике, $|V| = k$. Базовое множество $\mathcal{F}$ задается соотношениями (1)–(7). Тогда существуют вспомогательные задачи $\widetilde{I}_1 = \langle X, V, \widetilde{\rho}_1\rangle$ и $\widetilde{I}_2 = \langle X, V, \widetilde{\rho}_2\rangle$ и ИГ $U_1 \in \mathcal{U}(\widetilde{I}_1, \mathcal{F})$, $U_2 \in \mathcal{U}(\widetilde{I}_2, \mathcal{F})$ такие, что верны следующие оценки сложности:*

$$\begin{cases} Q(U_1) \preceq q^2 k^{1+\frac{2}{q}} \\ T'(U_1) \preceq q^2 \\ \widehat{T}'(U_1) \preceq q^2 \log_2 k \\ \frac{37}{1000} \leq \widehat{m}(I, \widetilde{I}_1) \leq \frac{38}{1000} \end{cases} \qquad \begin{cases} Q(U_2) \preceq q k^{1+\frac{1}{q}} \\ T'(U_2) \preceq q \\ \widehat{T}'(U_2) \preceq q \log_2 k \\ \frac{50}{1000} \leq \widehat{m}(I, \widetilde{I}_2) \leq \frac{51}{1000} \end{cases}$$

*где* $1 \leq q \leq \log_2 k$ *— натуральный параметр.*

**Литература**

1. Гасанов Э. Э., Кудрявцев В. Б. Теория хранения и поиска информации. — М.: ФИЗМАТЛИТ, 2002.

# БАЗА ЗНАНИЙ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ MPI-ПРИЛОЖЕНИЙ

**Дергунов А.В. (Нижегородский государственный университет им. Н.И. Лобачевского)**
*anton.dergunov@gmail.com*

Для анализа MPI программ часто используют программные системы, которые осуществляют сбор и визуализацию трассы выполнения программы. Но при использовании таких инструментов пользователь сталкивается с проблемой анализа больших объемов информации. Другой проблемой при использовании средств визуализации является то, что часто встречающиеся ситуации, приводящие к потерям производительности MPI программ, явно не визуализируются, т.е. отсутствует база знаний повышения производительности.

Поэтому возникает потребность в средствах, которые бы автоматизировали анализ трассы и подсказали пользователю, как повысить производительность его программы. В данной работе описывается программная система, выполняющая эту задачу.

Схема работы системы представлена на рис. 1. Исходными данными является трасса выполнения MPI приложения, полученная с помощью трассировщика. Она анализируется модулем автоматического анализа трассы. Результат анализа – список выявленных причин недостаточной производительности пользовательского приложения с указанием степени их влияния на общее время работы приложения.

Ключевым компонетом этой системы является база знаний причин недостаточных производительности MPI приложений, которая состоит из правил, описанных на специальном языке, разработанном в рамках этой системы. Правила используются в системе для следующих целей:

- описание составных событий, которые формируются на основе простых событий, собранных трассировщиком во время работы программы, или других составных событий, и представляют собой высокоуровневые операции программы;
- описание причин недостаточной производительности MPI приложений.

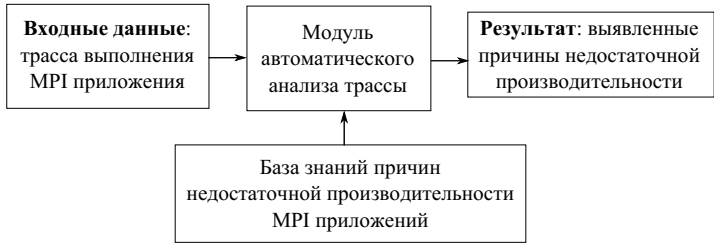



Рис. 1: Схема работы системы

Ниже приведен пример правила, описывающего событие операции приема данных, как состоящее из простых событий, соответствующих вызовам функций MPI_Recv_init и MPI_Start.

```
rule
    event e1
        type function_call_non_blocking_receive_operation
    event e2 type function_call_request_handling
    where
        e1.@function_name eq "MPI_Recv_init" and
        e2.@function_name eq "MPI_Start" and
        e1.request = e2.request
    compose event of type point_to_point_operation
        type = RECEIVE_OPERATION
        function_name = e1.funcion_name
        start_time = e2.start_time
        finish_time = e2.finish_time
        source = e1.source
        dest = e1.dest
        tag = e1.tag
        comm = e1.comm
    delete e1
    delete e2;
```

Одной из причин недостаточной производительности является плохая синхронизация приемов и передач данных в MPI програм-

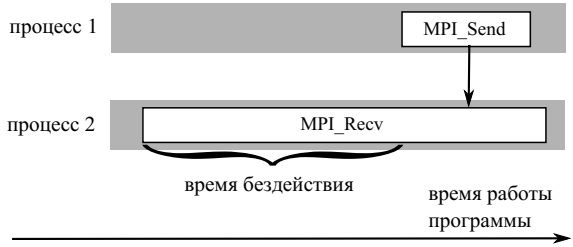



Рис. 2: Поздняя посылка данных

ме. В результате процедура приема данных может простаивать, дожидаясь посылки данных (см. рис. 2). Для решения этой проблемы нужно обеспечить, чтобы сообщения посылались как можно раньше, и таким образом повысить вероятность того, что сообщение прибудет до момента, когда оно потребуется другому процессу. Следующее правило описывает эту ситуацию:

```
rule
  event e type point_to_point_operation_group
  where
    e.receive_function_name eq "MPI_Recv" and
    e.send_start_time > e.receive_start_time
  create performance observation
    name = "Поздняя посылка данных"
    description = "Операция посылки данных вызывается
      позднее операции приема. Из-за этого блокирующая
      операция приема вынуждена простаивать."
    advice = "Улучшить синхронизацию приемов и передач
      данных, оправляя данные по возможности раньше, или
      использовать неблокирующие операции приема."
    impact = op1.send_start_time - op1.receive_start_time;
```

С помощью приведенного правила осуществляется обнаружение операций двухточечного обмена данными, операция приема данных в которых блокирующего типа (MPI_Recv) и время начала посылки данных позднее времени начала приема данных. При выполнении перечисленных условий система делает вывод о неэффективной

работе программы, связанной с поздней посылкой данных. В правиле указывается подробное описание этой причины недостаточной производительности и совет по ее устранению. Также вычисляется степень влияния на общее время работы программы, как разница между временем начала посылки данных и временем его приема.

База знаний системы состоит из правил, осуществляющих выявление следующих причин недостаточной производительности MPI программ: поздняя посылка данных, поздний прием данных, ранний прием данных при операции «от многих к одному» и т.д. Этот список был составлен на основе анализа литературы по оптимизации MPI программ и стандарта MPI. Более подробно состав базы знаний описан в работе [1].

В работе [1] описан эксперимент по повышению производительности MPI программы с помощью разработанной системы. В результате анализа трассы ее работы на кластере системой были установлены причины недостаточной производительности и выдан совет по изменению программы пользователя для улучшения ее производительности. После внесения изменений общее время работы программы существенно уменьшилось.

**Литература**

1. Карпенко С. Н., Дергунов А. В. Программные средства повышения производительности MPI-приложений // Супервычисления и математическое моделирование. Труды XII международного семинара. – Саров: ФГУП «РФЯЦ-ВНИИЭФ», 2011, с. 195-202.

## ГЕОМЕТРИЧЕСКИЙ ПОДХОД К РАСПОЗНАВАНИЮ ЗРИТЕЛЬНЫХ ОБРАЗОВ (КРАТКИЙ ОБЗОР)

**Козлов В.Н. (Москва, МГУ)**

Ниже содержательно описывается дискретно-геометрический подход к распознаванию зрительных и основные его результаты. Описание разбито на 5 пунктов.

I) Изображением здесь называется конечное множество точек в евклидовом пространстве. В частности, двумерным или плоским изображением называем конечное множество точек на плоскости. Содержательным обоснованием этому может служить то, что любое реальное (не цветное) изображение можно аппроксимировать изображением из точек, причем в нужной мере можно передать все градации «серого цвета» разной плотностью точек в разных частях изображения. Такое представление не закрывает дорогу и к рассмотрению цветных изображений, поскольку, как известно, цветное изображение можно представить тремя не цветными. Наконец, все, что мы видим, мы видим посредством глаз. Изображение из среды проецируется на сетчатку глаз, что приводит к возбуждению части рецепторных клеток, то есть, в конечном счете, к формированию на сетчатке аналога составленного из точек изображения.

Аналогично двумерному случаю, конечное множество точек в трехмерном пространстве может аппроксимировать с нужной степенью точностью предметы, сцены, и в целом окружающую среду. Наконец, конечные множества точек в четырехмерном пространстве могут представлять трехмерные сцены в динамике. При этом одна из осей интерпретируется как ось времени.

Итак, основной объект рассмотрения - конечные множества точек в евклидовых пространствах. С таким объектом можно уже работать формальными, математическими средствами, т.е. доказывать теоремы, делать количественные оценки, и пр. На этих теоремах можно основывать алгоритмы распознавания, базирующиеся уже не на правдоподобных рассуждениях (как при эвристическом подходе), а на точных доказательствах. Это то, что можно назвать доказательным, теоремным уровнем рассмотрения проблем распознавания изображений.

П) Итак, два изображения $A$ и $B$ - это два конечных множества точек, например, две фигуры, в значительной степени повторяющие

очертаниями, формой друг друга, и нам это обстоятельство нужно выяснить. Проблема, однако, в том, что априори нам неизвестно , что это за фигуры, более того, неизвестно, где у них правые-левые части, верх-низ, то есть нет ничего, кроме двух множеств точек. Как можно было бы определить схожесть по форме этих фигур? Если бы они были одинаковы, то, очевидно, это можно было бы выяснить наложением их друг на друга (например, движениями, т.е. изометрическими преобразованиями) так, чтобы они полностью (поточечно) совпали. Если же фигуры не вполне одинаковы, то надо движениями расположить их одну на другой так, чтобы степень их «несовпадения», «рассогласования» была бы наименьшей из возможных. В этом случае величина этого «рассогласования» была бы количественной характеристикой несовпадения по форме.

Сделаем это требуемое следующим образом. Для каждой точки каждого из изображений $A$ и $B$ в качестве характеристики рассогласования этой точки со сравниваемым изображением возьмем отрезок между нею и ближайшей точкой сравниваемого изображения. Тогда характеризовать степень несовпадения в целом данных двух изображений $A$ и $B$ будем наибольшим из отрезков, взятых по всем точкам обоих изображений. Обозначим длину этого отрезка через $\Delta(A, B)$.

Величина $\Delta(A, B)$ характеризует взаиморасположение конкретных множеств точек $A$ и $B$. Теперь будем преобразовывать $A$ и $B$, например, движениями так, чтобы они как можно более совпали, т.е. ищем такое взаиморасположение $A$ и $B$, при котором величина $\Delta(A, B)$ наименьшая из возможных.

Нетрудно видеть, что это задача на поиск экстремума: каждое из возможных положений для $A$ и $B$ характеризуется вполне определенной величиной $\Delta(A, B)$; надо построить множество всех возможных взаиморасположений для $A$ и $B$, порождаемых некоторыми их преобразованиями (движениями, например). Это даст множество $\{\Delta(A, B)\}$ величин $\Delta(A, B)$. Надо найти минимум на этом множестве (если он существует, конечно).

Проблема в том, что множество $\{\Delta(A, B)\}$ - бесконечное, более точно - континуальное по мощности. Таким оно будет, даже если взять в качестве преобразований для $A$ и $B$ наиболее простые преобразования - параллельного переноса. Естественно, проблема сохраняется, когда мы расширяем класс преобразований до изометрических - сочетаний параллельных переносов, поворотов и преобра-

зований симметрии относительно прямой. Следующий по сложности класс - преобразования подобия, когда добавляется возможность для $A$ и $B$ произвольно меняться в размерах. Наконец, в случае аффинных преобразований мы добавляем ко всему перечисленному еще и возможность для $A$ и $B$ сжиматься и растягиваться по произвольным направлениям.

Наиболее интересен с практической точки зрения класс аффинных преобразований. Ибо в этом случае мы сравниваем «по форме» два изображения $A$ и $B$ безотносительно к их положению на плоскости, поворотам, изменениям в размерах, сжатиям и растяжениям. Однако, напомним, мы не можем сравнивать $A$ и $B$ непосредственно перебором, попробовав все варианты взаиморасположений, ибо множество $\{\Delta(A,B)\}$ (точнее, его аналог для аффинного случая) - бесконечное.

Неожиданностью здесь было то, что, как теперь доказано, существует конечное подмножество взаиморасположений для $A$ и $B$, и, соответственно, конечное подмножество на $\{\Delta(A,B)\}$ , на котором нужный минимум только и может достигаться. Это конечное множество взаиморасположений $A$ и $B$ может быть построено. Это, в свою очередь, означает, что бесконечный и потому неосуществимый перебор всех возможных вариантов взаиморасположений $A$ и $B$ может быть заменен проверкой лишь конечного, заранее определенного числа их взаиморасположений. Таким образом, нет необходимости пробовать по разному смещать, поворачивать изображения $A$ и $B$ относительно друг друга, менять их размеры, и пр. - можно сразу идти туда, где только и может быть решение.

Ш) Одна из особенностей зрительной информации - ее огромные объемы [1,2]. Глаз человека содержит около 130 миллионов светочувствительных элементов (палочек и колбочек). Вместе с тем распознавание изображений осуществляется иногда лишь за доли секунды. Вряд ли это можно сделать, используя целиком всю информацию на сетчатке глаза. Должно существовать некоторое «сито» и для изображений из среды, и для визуальной информации из памяти при оперировании с ними. Парадокс, однако, состоит в том, что попытки до собственно распознавания выделить на изображениях «опорные точки» , «важные детали» и пр. (как правило, средствами эвристики), есть тоже распознавание, т.е. возникает до некоторой степени замкнутый круг. Ясно, что нужно избегать этого.

Зададимся некоторым числом $r$ ($r \geq 0$). Выберем на изображении $A$ подмножество $A^r$ точек из $A$ с тем условием, чтобы каждая точка из $A$ находилась бы на расстоянии, не большем $r$, от какой-либо точки из $A^r$. Ясно, что $A^r$ можно трактовать как подизображение изображения $A$, в частном случае $A^r$ может совпадать с $A$. Изображение $A^r$ можно интерпретировать как покрытие $A$ кругами радиуса $r$ с центрами в точках из $A^r$ , причем такое, что каждая точка исходного изображения попадает хотя бы в один из этих кругов.

Содержательно $A^r$ трактуется как «огрубление» изображения $A$, устранение на нем излишних деталей и подробностей, поэтому $A^r$ называем эскизом изображения $A$. При заданном параметре $r$ существует некоторое множество $\{A^r\} = \{A^r_1, \ldots, A^r_k\}$ эскизов, причем исходное изображение $A$ входит в это множество. Однако интерпретировать как «огрубление» $A$ можно, очевидно, лишь те эскизы из $A^r$,которые состоят из меньшего числа точек, чем исходное изображение $A$. В частности, в $\{A^r\}$ есть изображение (может быть, не одно) с наименьшим среди $A^r_1, \ldots, A^r_k$ числом точек. Такое изображение назовем остовом изображения $A$.

При оценке похожести двух изображений достаточно, может быть, оценить их сходство в целом, в главных чертах, без деталей. Такого рода интуитивные соображения неявно предполагают, что вместо собственно изображений будут использованы другие изображения, полученные из исходных устранением излишних деталей, то есть в нашем случае - эскизы. Оказалось возможным теоремным образом связать схожесть между эскизами со схожестью между оригиналами. Это дает возможность, имея дело с эскизами, делать оценки для схожести исходных изображений.

IV) Код изображения $A$ - пара $< M_A, T_A >$, где $M_A$ - множество номеров точек. Множество $T_A$ состоит из чисел с индексами вида $\rho_{mnu,ksp}$ . Здесь $\rho_{mnu,ksp}$ получается отношением площадей треугольников с вершинами в точках изображения с номерами соответственно $m, n, u$ и $k, s, p$ (естественно при условии, что площадь треугольника с вершинами в точках $k, s, p$ не равна нулю). Такие числа $\rho$ строятся для всех пар треугольников. Подчеркнем, что только на этапе формирования числа $\rho_{mnu,ksp}$ мы говорим о треугольниках и о площадях - далее этой информации нет и $T_A$ состоит только из чисел с индексами.

Доказано, что коды двух изображений одинаковы (с точностью

до перенумерации точек) тогда и только тогда, когда эти изображения переводятся одно в другое аффинным преобразованием. Тем самым, этот код задает изображение с точностью до аффинных преобразований. Это означает, что по коду изображение можно восстановить по разному, но все варианты восстановления будут отличаться друг от друга только аффинными преобразованиями.

Аналогичным свойством обладает код $< M_A, T_A >$ для точек в трехмерном пространстве (трехмерные изображения или тела). Различие с двумерным случаем состоит только в том, что числу $\rho$ приписаны две четверки индексов $m, n, u, v$ и $k, s, p, q$ и само число получается отношением объемов тетраэдров с вершинами в соответствующих точках. Этот код тоже описывает тело с точностью до аффинных преобразований.

Доказано, что аналогичные коды существуют и для конечных множеств точек в пространствах размерности $n$, где $n > 3$.Аналогичные коды построены и для проективных преобразований [2].

V) Пусть теперь есть некоторое тело $Q$ (трехмерное изображение) и двумерное изображение $S$. Вопрос , который нас интересует, может быть поставлен так: как соотносятся $Q$ и $S$, если известно, что $S$ является проекцией тела $Q$ на некоторую плоскость.

Эта задача является основой для восстановления трехмерного изображения по его плоским проекциям, что в свою очередь есть важнейшая составляющая моделей стереовосприятия для роботов, живых организмов и процедур построения изображений в томографии.

В целом рассмотрения такого рода для стереовосприятия приводят к следующей задаче. Имеются плоские проекции $S_1$ и $S_2$ тела $Q$ (проекции на сетчатки). Однако в нашем распоряжении есть только плоские изображения $\tilde{S}_1$ и $\tilde{S}_2$, полученные из соответственно $S_1$ и $S_2$ произвольными аффинными преобразованиями (какими - неизвестно). Кроме того , поточечное соответствие между $\tilde{S}_1$ и $\tilde{S}_2$ не задано. Надо восстановить тело $Q$.

Ясно, что в такой постановке снимается и вопрос о расстоянии между сетчатками, и о направлении лучей проекции, которыми получены точки на $S_1$ и $S_2$ .

Оказывается, в такой постановке задача решаема: по $\tilde{S}_1$ и $\tilde{S}_2$ тело $Q$ восстанавливается с точностью до аффинных преобразований, т.е.

строится тело $\tilde{Q}$, отличающееся от исходного $Q$ некоторым аффинным преобразованием. При этом определяется и поточечное соответствие между $\tilde{S}_1$ и $\tilde{S}_2$. Это не эмпирический результат - соответствующая теорема доказывается.

К настоящему времени есть несколько компьютерных реализаций изложенного подхода.

1) Распознавание произвольных черно-белых фигур: цифр, букв, подписей, иероглифов, рисунков, и пр. На экране «мышкой» рисуются образцы фигур, которые нужно распознавать, они отправляются в память. Распознаваемый объект тоже рисуется на экране. Распознавание состоит в выборе того объекта из памяти, который лучше всего «натягивается» на объект на экране методами, описанными кратко в пунктах Ш и IV. Это наилучшее «натяжение» выводится на экран и является ответом. Распознавание не зависит от размеров, ориентации, положения на плоскости, сжатий-растяжений и локальных особенностей сравниваемых фигур.

Есть более сложный вариант программы, когда на экране - несколько фигур, возможно пересекающихся, наложенных друг на друга. Образцы из памяти должны сами «найти» на экране те фигуры, на которые они лучше всего «натянутся».

2) Восстановление трехмерных изображений по стереопарам (стереофотографиям). Исходными являются две фотографии объекта или сцены в двух несколько разных ракурсах (обычные стереопары). Методами, изложенными кратко в пункте VI, восстанавливается трехмерное изображение («виртуальное», в памяти компьютера). Это дает возможность получать новые проекции, т.е. объект в новом ракурсе (фактически новые изображения объекта). Если это делать последовательно, то на экране объект или сцена поворачиваются так, как если бы мы перемещались относительно них (конечно, в определенных пределах: восстанавливаются только те точки сцены, которые присутствуют на обеих исходных проекциях).

3) Распознавание мелодий. Мелодии, заносимые в память и распознаваемые - только из «чистых» , «синусоидальных» звуков (они сравнительно легко создаются непосредственно на компьютере). Такие мелодии легко «визуализируются» в виде двумерной картинки. Распознаваемая мелодия может отличаться от записанной в память громкостью, темпом, тональностью.

**Литература**

1. Козлов В.Н. Введение в математическую теорию зрительного восприятия.М.: Издательство Центра прикладных исследований при механико-математическом факультете МГУ, 2007. 136 стр.

2. Алексеев Д.В. Кодирование изображений, инвариантное относительно проективных преобразований., Интеллектуальные системы, т.13, вып. 1–3, М. 2009, стр.35–40.

# УНИВЕРСАЛЬНАЯ МОДЕЛЬ АЛГОРИТМОВ КОЛЛЕКТИВНОГО РАЗУМА И ЕЕ РЕАЛИЗАЦИЯ

**Конончук Д.О., Окуловский Ю.С.**
**(Уральский государственный университет им. А.М. Горького)**
*kononchukdmitry@gmail.com*

Алгоритмы коллективного разума (АКР) являются в настоящее время одной из самых динамично развивающихся отраслей алгоритмов искусственного интеллекта. Они основаны на т.н. «интеллекте толпы» – явлении, при котором система из множества слабо интеллектуальных, равноправных и децентрализованных сущностей проявляет свойства интеллектуальности. Примерами таких алгоритмов являются «муравьиные алгоритмы» [1], метод роя частиц [2], искусственные иммунные системы [3], и т.д.

В отличие от других интеллектуальных систем (нейронные сети, экспертные системы, генетические алгоритмы), для АКР до сих пор не существует общеизвестной системы поддержки, решающей то множество сервисных задач, которое возникает при программной эмуляции подобных алгоритмов. В связи с этим, в каждой реализации того или иного АКР приходится заново решать проблемы распараллеливания вычислений, ввода и вывода данных из системы, ее отладки и многие другие. Это усложняет процесс разработки и делает практически невозможной интеграцию различных решений. Кроме того, из-за отсутствия единого фреймворка производительность различных алгоритмов сложнее сравнивать: время работы и потребление памяти разных решений будут в первую очередь зависеть не от принципиальных различий алгоритмов, а от особенностей их реализации. В некоторых случаях в качестве фреймворка могут быть использованы т.н. системы агент-ориентированного моделирования (см. например, [4]). Но, к сожалению, эти системы не поддерживают многих свойств, являющихся существенными для АКР. Кроме того, среди них нет открытых решений на языке C#. Таким образом, для облегчения реализации, интеграции и сравнения различных типов и вариаций АКР требуется создание универсальной системы поддержки алгоритмов данного типа. Эта система должна подходить для создания на ее базе эмуляций всех общеизвестных АКР и, кроме того, быть удобной для использования, изучения и расширения (поскольку возможно применение ее также и для поддержки других систем, схожих с АКР).

На основании анализа и классификации существующих алгоритмов коллективного разума, была построена их общая модель. Основным элементом нашей модели АКР являются *агенты*. Муравьи в муравьиных алгоритмах и клетки крови в искуственных иммунных системах являются агентами. С точки зрения программиста, агенты представляют собой объекты различных классов, которые поддерживают некоторое количество методов, оформленных в интерфейсе. Помимо агентов, есть также *мир*, который содержит агентов и дополнительную информацию о них. Время в мире дискретизировано тактами, за каждый такт все агенты делают *ходы* последовательно. Также спроектирована система транзакций, позволяющая добиться независимости агентов друг от друга и иллюзии параллельности их работы. Эта система ограничивает свободу программирования логики в действиях агентов, допуская лишь такие действия, как перемещение агентов, их смерть и появление, а также локальное изменение характеристик мира. Указанных действий, однако, достаточно для большинства алгоритмов коллективного разума. Система транзакций также представляет собой эффективный механизм для распараллеливания алгоритмов.

Пространство в мире дискретизировано *ячейками*, в каждой ячейке может содержаться один или множество агентов, в зависимости от настроек алгоритма. Также в каждой ячейке может хранится произвольная структура данных, которая представляет доступную информацию о мире. Эта информация может быть изменена агентами, а также изменяться самопроизвольно, с течением времени. На ячейках задана *топология*, которая определяет для каждой ячейки перечисление соседних с ней ячеек. Это позволяет легко реализовывать локальное зрение агентов: при совершении хода агент, как правило, учитывает состояние лишь несколько соседних ячеек, соответственно, ему достаточно лишь пройти по предоставляемому топологией перечислению. Разработаны топологии для двумерных и трехмерных сеток, а также топология для произвольного графа.

Агенты имеют следующие возможности взаимодействия друг с другом. Во-первых, они могут обмениваться информацией через структуры данных мира, изменяя их в рамках своих ходов. Такой подход наиболее характерен для муравьиных алгоритмов. Вторым способом взаимодействия является *эфир*, в рамках которого сообщения агентов становятся доступными остальным агентам на следу-

ющей итерации. Эфир может быть использован при моделировании иммунных систем при моделирования выброса клетками сигнальных веществ. Третьим способом взаимодействия является произвольное изменение агентами параметров друг друга, вызов методов и т.д. Данный способ взаимодействия несовместим с системой транзакций.

Приведенная модель, с одной стороны, удобна для программной реализации, и, с другой стороны, позволяет легко описывать АКР любого типа. Была создана последовательность прототипов и, затем, полноценная реализация этой модели, предоставляющая множество удобных инструментов для описания АКР. На каждой итерации разработки производились различные типы тестирования, результаты которых использовались для дальнейшего усовершенствования продукта. Система реализована на языке C# 4, под платформу .Net 4.0, совместима с платформой Mono. В ходе разработки активно использовались технологии контрактного и параллельного программирования.

На основании данной модели были реализованы известные алгоритмы из класса муравьиных алгоритмов, искусственных иммунных систем и роя частиц. В настоящее время нами ведутся работе по реализации других алгоритмов коллективного разума на базе разработанной системы. Кроме того, созданная модель применяется для создания интеллектуальных алгоритмов в компьютерных играх, а также для моделирования социума.

**Литература**

1. Dorigo M. Optimization, Learning and Natural Algorithms // PhD thesis. – Politecnico di Milano, Italie, 1992.

2. Kennedy J., Eberhart R. Particle Swarm Optimization // Proceedings of IEEE International Conference on Neural Networks IV. – 1995, pp. 1942 - 1948.

3. Dasgupta D. (editor) Artificial Immune Systems and Their Applications // – Springer-Verlag, Inc. Berlin, January 1999.

4. Axelrod R. The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration // – Princeton: Princeton University Press, 1997.

## ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ НА REFAL "HAPPY REFAL"

**Котельников С.В. (Ярославский государственный университет им. П.Г.Демидова (ЯрГУ))**
*E-mail: svkotelnikov@gmail.com*

Актуальной задачей обучения в современных условиях является индивидуализация, которая обеспечивается использованием компьютера. Данная работа представляет собой интеллектуальную систему обучения языку логического программирования REFAL(в объеме REFAL-2).

Целью работы является разработка программной системы обучения языку REFAL, позволяющей предоставлять изучаемый материал, контролировать его освоение, как в части понимания материала, так и в части его практического использования.

В качестве системы обучения, отвечающей поставленным целям, была разработана программная система, реализующая стратегию поэтапного обучения, которая не только организует активное изучение материала, но и контролирует, а так же направляет процесс обучения. Обучение представлено тремя последовательными этапами: на первом этапе происходит изучение теоретического материала и тестирование его усвоения, второй этап обучения представлен выполнением упражнений, требующих осмысленного понимания материала, и последний этап, являющийся основным, представлен применением полученных знаний и навыков для написания полноценных программ на языке REFAL.

Система предоставляет пользователю ресурсы в зависимости от его уровня доступа. Незарегистрированный пользователь получает только возможность просмотра методических материалов и сравнительного состояния обучения студентов. Заметим, что сервер данной интеллектуальной системы предоставляет открытые для общего пользования веб-сервисы, позволяющие получить информацию о состоянии обучения студентов, что может быть использовано для размещения данной информации на сайте в интернете или на каком либо клиентском приложении, что делает обучение прозрачным для преподавателей, студентов и заинтересованных лиц. Зарегистрированный пользователь, не являющийся администратором, помимо возможностей незарегистрированного пользователя, также получает возможность прохождения обучения и возможность пользоваться

интерпретатором REFAL. Наконец, пользователь с правами администратора имеет возможность изменять параметры системы, редактировать задания, учетные записи пользователей, сессии обучения и т.д. Рассмотрим подробно каждый из этапов обучения, которое предлагается пройти зарегистрированным в системе пользователям.

На первом этапе обучения студенту предоставляется теоретический материал и после его изучения студент переходит к стадии тестирования понятий материала. При этом случайным образом выбираются вопросы теста с закрытыми вариантами ответа, для решения которых требуется указать все правильные ответы. В случае правильного ответа на вопрос теста, количество вопросов теста уменьшается и студент получает возможность вернуться к изучению теории или перейти к ответу на следующий вопрос, а в случае неправильного ответа количество вопросов в тесте увеличивается и студент отсылается к той части теоретического материала, на основе которой был составлен вопрос. При превышении предельного количества ошибок, сеанс обучения заканчивается без сохранения прогресса обучения. При правильных ответах на все предлагаемые вопросы теста теоретической части обучения, происходит переход к следующему этапу обучения – этапу выполнения упражнений требующих осмысленного понимания материала.

На втором этапе обучения также случайным образом студенту предоставляется ряд заданий. Выполнение каждого из таких заданий состоит в написании в окне редактора некоторой части кода, проверяющей осмысленное понимание материала. Задания этого этапа обучения представлены написанием отдельных, синтаксически и семантически правильных, функций на языке REFAL. При неправильном решении задания студенту указывается ошибка, и он вновь отсылается к теоретическому материалу так же, как и на предыдущем этапе обучения. Число упражнений на этом этапе также растет в случае неправильных ответов и при превышении порога неправильных ответов сессия завершается. Только после правильных ответов на все предлагаемые упражнения данной части обучения студент может перейти к главному этапу обучения.

Последний этап обучения представлен этапом обучения разработке программ на языке REFAL и является основной целью данной обучающей системы. На этапе обучения разработке программ в качестве заданий предлагается писать программы на REFAL по

условиям, сформулированным в заданиях. Помимо кода программы предоставляется возможность написания набора тестов к программе, с возможностью их пошагового выполнения. Тесты представляют собой набор ожидаемых результатов выполнения программы для заданных входных значений. Пошаговое выполнение программы является очень важной частью обучения, позволяющей наглядно проследить, как происходит выполнение REFAL-программы и совпадает ли оно с ожидаемым. Данная работа дает возможность применить студенту приобретенные в процессе обучения знания и навыки при написании программ ня языке REFAL. Помимо пользовательских тестов существует "полный" набор тестов прикрепленных к задаче, не отображаемых интерфейсом, и по результату выполнения которых система определяет правильность написанного пользователем программного кода; в случае логической ошибки система выдает соответствующее сообщение, в котором указываются тесты на которых была обнаружена ошибка. В случае превышения предельного количества логических ошибок в задании, задание меняется и количество заданий предлагаемых к решению увеличивается. При превышении предельного количества неправильно решенных заданий сеанс обучения завершается.

Настройка системы обучения должна производиться при помощи выбора значений ее параметров во время опытной эксплуатации. К таким параметрам относятся: исходное количество заданий на каждом этапе обучения, количество заданий на которые уменьшается или возрастает количество предлагаемых заданий на отдельных этапах обучения при правильных и неправильных решениях соответственно, предельное допустимое количество ошибок на каждом из этапов обучения и т.д.

Система реализована на базе платформы Java EE с использованием ORM баз данных, а также различных сторонних Java и Java Script библиотек. Работа пользователей в данной системе осуществляется через web-интерфейс. Помимо пользовательских интерфейсов система предоставляет ряд веб-сервисов, что позволяет ей взаимодействовать с другими внешними системами и клиентами. Заметим, что в связи с отсутствием подходящих сторонних программных продуктов производящих пошаговую интерпретацию программ REFAL и их анализ, интерпретатор и синтаксический и семантический анализаторы написаны в рамках данной программной системы.

В 2011 г. данная работа была представлена на XIX Международной студенческой конференции-школе-семинаре “Новые информационные технологии”, где была удостоена Диплома III степени за лучшую научную работу представленную на конференции. Автор признателен своему научному руководителю профессору В.С.Рублёву за постоянное внимание к работе и чуткое руководство.

**Литература**

1.Базисный Рефал и его реализация на вычислительных машинах. – М.: ЦНИПИАСС, 1977. – 258 с.

2. Климов, А. В. Система программирования Рефал-2 для ЕС ЭВМ. Описание входного языка / Ан. В. Климов, С. А. Романенко. – М.: ИПМ им. М. В. Келдыша АН СССР, 1987. – 52 с.

3. Романенко, С. А. Реализация Рефала-2 / С. А. Романенко. – М.: ИПМ им. М. В. Келдыша АН СССР, 1977. – 191 с.

4. Турчин, В. Ф. Феномен науки: кибернетический подход к эволюции / В. Ф. Турчин. – М.: ЭТС, 2000. – 368 с.

# МЕТОД ОРГАНИЗАЦИИ РАБОТЫ С ДАННЫМИ В ПРИКЛАДНЫХ СИСТЕМАХ РАСПОЗНАВАНИЯ ОБРАЗОВ

**Максимова А.Ю. (Донецк, Институт прикладной математики и механики НАН Украины)**
*Maximova.Alexandra@mail.ru*

При разработке информационных систем часто возникает задача классификации, которая решается методами теории распознавания образов и анализа данных. В готовых системах распознавания образов (СРО), таких как «Распознавание 1.0» и STATISTICA реализован ряд известных методов, однако без участия специалиста выбрать подходящий и оценить качество полученного решения достаточно сложно [1]. Другим недостатком таких систем является организация работы с данными по средствам работы с файлами следующего формата: каждая строка определяет объект $\omega$ из множества всех объектов $W$ в виде вектора значений его признаков. Для задач обучения с учителем также определяется номер класса образов, к которому относиться объект $\omega$. Такая таблица называется выборкой прецедентов. Иногда приходится подготавливать две выборки прецедентов — обучающую и контрольную. При разработке крупных программной системы, внедряемых на производстве, процесс формирования выборок прецедентов необходимо автоматизировать учитывая особенности хранения и работы с данным в конкретной предметной области.

В работе предлагается метод организации работы с данными для системы распознавания образов. Структурная схема работы с данными приводится на рис. 1. На схеме присутствуют три формы представления одной и той же информации.

Изначально данные реального мира представлены в неформализованном виде: знания оператора, показания датчиков либо уже сохраненные файлы практически любых форматов. Второй формой представления данных является БД предметной области, основанная не реляционной модели данных, которая на данном этапе развития информационных систем получила широкое практическое применение. В ней, наряду с актуальными для задачи классификации данными, сохраняется и вся другая информация предметной области. Третья форма представления данных адаптирована для работы алгоритмов распознавания образов и хранит в виде таблиц коллекции обучающих выборок, так как в теории распознавания образов и анализа данных сформировалась практика представления обучающих

выборок в виде стратифицированных таблиц.


Рис. 1: Структурная схема работы с данными

Программный модуль «Анализатор» является интерфейсом между данными реального мира и БД предметной области. Он является уникальным для каждой конкретной прикладной задачи.

Программный модуль «Интерпретатор» является интерфейсом между данным предметной области и системой распознавания образов. Его основной функцией является предоставление всех необходимых данных для СРО. Данный модуль является универсальным для любой предметной области. Настройка его работы осуществляется с помощью пакета специальных правил, определенных для каждого задания СРО. Правила содержат информацию, позволяющую сформировать структуру классов образов и определить систему информационных признаков на основании БД предметной области. Доступ к объектам БД выполняется по шаблону «имя отношения». «имя атрибута». Также применяются ограничения на значения конкретных атрибутов. Возможно использование разных правил для формирования обучающих и тестовых выборок.

**Основные функции модуля «Интерпретатор».**

В системе распознавания образов пользователь дает команду на

выполнения определенного задания. Для каждого задания известен метод принятия решения, базирующийся на определенном алгоритме распознавания и способе контроля качества полученного решения. Для каждого метода принятия решения задан свой набор правил, который понимает модуль «Интерпретатор».

Предлагаемая методика организации прикладных систем распознавания образов позволяет создавать контекстно-независимые системы рапознавания образов, которые могут быть интегрированы в существующую информационную систему по средствам настройки модуля «Интерпретатор». Другим достоинством данного подхода является возможность добавлять в программную систему новые типы заданий, не меняя при этом структуру БД предметной области. Следует отметить также поддержку иерархической структуры классов образов и поддержку как пакетного режима, так и on-line режимов обработки данных.

Предлагаемый метод организации работы с данными использован при разработке автоматизированной системы контроля качества нефтепродуктов[2].

**Литература**

1. Журавлев Ю. И., Рязанов В. В., Сенько О. В. Распознавание. Математические методы. Программная система. Практические применения. –М.: ФАЗИС, 2006, 176 с.

2. Козловский В. А., Максимова А. Ю. Нечеткая система распознавания образов для решения задач классификации жидких нефтепродуктов. // Научные работы ДонНТУ, серия «Информатика, кибернетика и вычислительная техника». - 2011, - №13 (185), - С. 200-205.

# ПРИМЕНЕНИЕ СЕМАНТИЧЕСКОГО ГРАФА ДЛЯ РЕШЕНИЯ ТЕКСТОВЫХ ЗАДАЧ

**Перпер Е.М. (МГУ им. М.В.Ломоносова)**
*e_m_perper@mail.ru*

В работе рассматривается метод автоматического решения текстовых задач с помощью проведения преобразований над семантическими графами текста задачи. При этом каждое преобразование соответствует определённому шагу решения задачи человеком.

**Введение**

Одним из ключевых при решении задачи обработки естественного языка является понятие смысла, которое достаточно трудно чётко определить. Одна из частных проблем обработки естественного языка — автоматическое решение текстовых задач — имеет то преимущество, что понятие "смысл" здесь определить легко. Смысл задачи выражается математическими действиями, которые необходимо произвести для нахождения её решения. Для того, чтобы представить текст задачи в виде чего-то более понятного для машины, чем просто набор слов, следует провести некоторый анализ этого текста. Как правило, выделяют три последовательных этапа анализа: морфологический, синтаксический и семантический. В методе автоматического решения текстовых задач, предложенном в [1] и реализованном А.С.Подколзиным на практике, результатом семантического анализа является набор логически формализованных утверждений, передающий смысл задачи. Обычно же семантический анализ завершается построением семантического графа, выражающего смысловые отношения между частями текста. Если семантические графы предложений задачи построены, то решение задачи можно свести к цепочке действий над этими графами.

**Основные понятия и формулировка результатов.**

Уже упоминавшийся в данной работе семантический анализ — это анализ, который выявляет семантическую структуру предложения, состоящую из сем.узлов и сем.отношений. Семантический узел — это слово, устойчивое словосочетание ( например, "не хватило духа" ) или жёсткая синтаксическая группа ( "тридцать три" ). Семантическое отношение - это некая универсальная связь между семантическими узлами, усматриваемая носителем языка в тексте. В системе русско-английского машинного перевода ДИАЛИНГ ( см. [2]) семан-

тическое отношение описывается как R(A,B), где A — зависимый член отношения, B — управляющий тип отношения, что означает: "A является R для B" . Например, семантическое отношение, соответствующее словосочетанию "сумка из кожи" , запишется как "матер(кожа, сумка)" , а словосочетанию "жить в глуши" соответствует запись "лок(глушь, жить)" . Если считать семантические узлы вершинами графа, а семантические отношения — рёбрами графа, то результат семантического анализа предложения можно представить в виде нагруженного графа, называемого семантическим графом. В данной работе предложена идея решения текстовых задач путём действий над семантическими графами предложений текста задачи. Приведён пример решения простой задачи предложенным методом.

**Решение текстовых задач с помощью преобразования семантических графов.**

В каждой текстовой задаче содержится предложение, из которого мы узнаём, что же требуется от человека, решающего задачу. Вышеуказанное требование выражено местоимением (например, "сколько "какое"и т.д.). Если в семантическом графе данного предложения (будем называть этот граф графом-вопросом задачи) заменить это местоимение на число, соответствующее ответу задачи, то полученный граф будет семантическим графом предложения, являющегося ответом задачи (будем называть его графом-ответом задачи). Граф-ответ задачи можно получить из графов предложений текста задачи, подвергнув их определённым логически обоснованным изменениям. Фактически, каждый этап решения человеком задачи можно свести к преобразованиям семантических графов предложений текста задачи.

Рассмотрим простую задачу и соответствующие ей семантические графы.

Задача. У Миши 2 яблока, а у Кати — на 5 яблок больше. Сколько яблок у Кати?

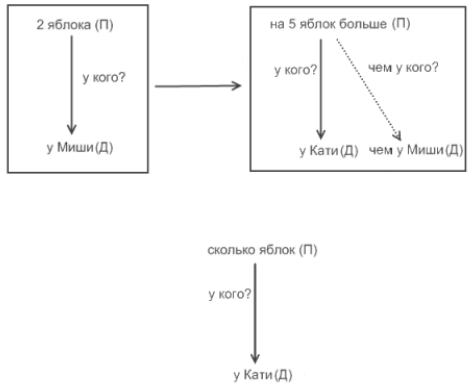



Рис. 1. Семантические графы задачи

Обратим внимание на последний граф на рис.1.: почти так и должен выглядеть граф ответа, лишь вместо "сколько" должно находиться конкретное число. На первом шаге решения задачи можно, пользуясь графом "у Миши — 2 яблока" заменить условия "чем у Миши" на "чем 2 яблока" .

Это можно было бы записать в виде правила, которое можно было бы применять во всех аналогичных случаях: "[на N единиц объекта A больше, чем у субъекта S]; [M единиц объекта A у субъекта S] $\Longrightarrow$ [на N единиц объекта A больше, чем M единиц объекта A]" . Словам "у Миши" , вообще говоря, не соответствует какое-либо числовое значение, в то время как словам "2 яблока" — соответствует. Именно поэтому мы осуществляем замену (не противоречащую смыслу). Теперь мы можем произвести необходимые вычисления. Правило можно записать так: "[на N единиц объекта A больше]+[чем M единиц объекта A] = [M+N единиц объекта A]" . После применения этого правила граф будет совпадать с графом-вопросом задачи, у которого вместо "сколько" находится число 7. Это и есть граф-ответ. Задача, таким образом, будет решена. Если задать определённый набор правил, соответствующих приёмам, используемых человеком при решении задачи, то задачу можно будет решать автоматически.

Это можно устроить, например, так: для конкретного правила проверять, применимо ли оно ли оно к семантическим графам задачи в данный момент, и если применимо, преобразовывать графы задачи по этому правилу; если же это правило неприменимо, перейти к следующему правилу. Если одновременно применимо несколько правил, можно рассмотреть каждый вариант по отдельности.

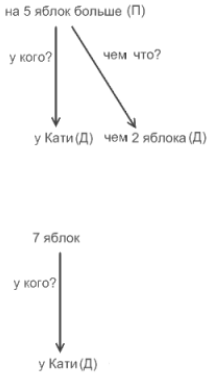



Рис. 2. Графы задачи после всех преобразований. Второй сверху — граф-ответ.

Таким образом, если в процессе решения задачи не происходит перехода к решению уравнений, описанный выше метод выглядит вполне приемлемым. Решению же уравнений довольно трудно сопоставить преобразование семантических графов, и для таких случаев требуются специальные методы решения.

Автор выражает благодарность профессору Эльяру Эльдаровичу Гасанову за научное руководство и помощь в подготовке статьи.

**Литература**

1. Подколзин А. С. Компьютерное моделирование логических процессов. — М.: Физматлит, 2008.

2. Сокирко А. Семантические словари в автоматической обработке текста (по материалам системы ДИАЛИНГ).
http://www.aot.ru/docs/sokirko/sokirko-candid-4.html.

# СУММИРОВАНИЕ ПО ПОЛУГРУППОВОЙ ОПЕРАЦИИ ДЛЯ ДВУМЕРНОЙ ЗАДАЧИ ИНТЕРВАЛЬНОГО ПОИСКА С ФИКСИРОВАННОЙ СТОРОНОЙ

**Пивоваров А.П. (МГУ им. М. В. Ломоносова)**
*pivizz@gmail.com*

Двумерная задача интервального поиска с фиксированной стороной рассматривалась ранее, например в [1], однако там рассмотрен её перечислительный вариант. Здесь мы рассматриваем такой вариант этой задачи, в котором каждой точке базы данных приписан некий «ранг» – элемент некоторой коммутативной полугруппы. Задача состоит в том, чтобы для запроса, задающего ограничения сверху и снизу для одной координаты и только ограничение сверху для другой, найти результат полугрупповой операции на рангах всех тех и только тех элементов базы, которые удовлетворяют запросу. В качестве коммутативной полугруппы может, например, выступать, некоторое множество чисел с операцией взятия максимума.

В работе [2] рассматривалась задача, подобная данной, но основанная на двумерной задаче о доминировании. Были получены несколько семейств алгоритмов и с их помощью оценена функциональная сложность соответствующей задачи. Двумерная задача о доминировании является частным случаем двумерной задачи интервального поиска с фиксированной стороной. Таким образом, здесь мы рассматриваем задачу, частный случай которой разобран в [2].

Сформулируем результаты данной работы.

Понятия *вычислительной задачи информационного поиска (ВЗИП)* и *вычислительного информационного графа (ВИГ)* совпадают с предложенными в [3] и используемыми позднее в работе [2].

Пусть $\mathcal{Z} = (Z, \bigoplus)$ – коммутативная полугруппа с нулём. Операцию $\bigoplus$ в дальнейшем будем для простоты именовать *суммой*. Рассмотрим тип вычислительных задач информационного поиска $S^{\mathcal{Z}}_{int2^*} = \langle X_{int2^*}, Y^{\mathcal{Z}}_{int2^*}, Z, \rho_{int2^*}, \xi^{\mathcal{Z}}_{int2^*} \rangle$, где $X_{int2^*} = \{(x'_1, x''_1, x_2) \in [0,1]^2 \colon x'_1 \le x''_1\}$, $Y^{\mathcal{Z}}_{int2^*} = [0,1]^2 \times Z$, отношение поиска $\rho_{int2^*}$ определено таким образом, что для любых $x = (x'_1, x''_1, x_2) \in X_{int2^*}$ и $y = (y_1, y_2, y_*) \in Y^{\mathcal{Z}}_{int2^*}$ соотношение $x \,\rho_{int2^*}\, y$ справедливо тогда и только тогда, когда одновременно выполнены условия $x'_1 \le y_1 \le x''_1$ и $y_2 \le x_2$. Элементы множества записей $Y$ можно рассматривать как

точки на плоскости, которым дополнительно приписаны веса из $Z$. Функция ответа $\xi^{\mathcal{Z}}_{int2^*} : 2^Y \to Z$ определена на конечных подмножествах множества $Y$ следующим образом (значение на бесконечных подмножествах $Y$ определять не требуется): пусть $V = \{y^1, \ldots, y^k\} \subset Y$ и $y^i = (y^i_1, y^i_2, y^i_{\mathcal{Z}})$, тогда $\xi^{\mathcal{Z}}_{int2^*}(V) = y^1_{\mathcal{Z}} \bigoplus \cdots \bigoplus y^k_{\mathcal{Z}}$. Отдельно определим $\xi^{\mathcal{Z}}_{int2^*}(\varnothing)$ равным нейтральному элементу $\mathcal{Z}$.

Вычислительный информационный граф (ВИГ) для решения задач введённого выше типа $S^{\mathcal{Z}}_{int2^*}$ будем строить над базовым множеством $\mathcal{F}^{\mathcal{Z}}_{int2^*} = \langle F_{int2^*}, G_{int2^*}, H_{\mathcal{Z}}, M_{\mathcal{Z}}, m^{\mathcal{Z}}_0, \sigma_{id}\rangle$. Положим $F_{int2^*} = \left\{f^i_a \colon a \in [0,1], i \in \{1', 1'', 2\}\right\}$, где для любого $x = (x'_1, x''_1, x_2) \in X_{int2^*}$ значение $f^{1'}_a(x)$ равно 1 тогда и только тогда, когда $x'_1 \le a$; значение $f^{1''}_a(x)$ равно 1 тогда и только тогда, когда $x''_1 \ge a$; значение $f^2_a(x)$ равно 1 тогда и только тогда, когда $x_2 \ge a$. Предикат $f^2_0$, значение которого равно 1 на любом запросе $x$, будем называть *тождественно единичным предикатом*. Множество переключателей $G_{int2^*} = \left\{g^i_a \colon a \in [0,1], i \in \{1', 1'', 2\}\right\}$, где для любого запроса $x = (x'_1, x''_1, x_2) \in X_{int2^*}$ имеет место $g^{1'}_a(x) = \begin{cases} 1, & \text{если } x'_1 > a; \\ 2, & \text{если } x''_1 \le a \end{cases}$; $g^{1''}_a(x) = \begin{cases} 1, & \text{если } x''_1 < a; \\ 2, & \text{если } x''_1 \ge a \end{cases}$; $g^2_a(x) = \begin{cases} 1, & \text{если } x_2 < a; \\ 2, & \text{если } x_2 \ge a \end{cases}$. Множество состояний ВИГ $M_{\mathcal{Z}}$ совпадает со множеством элементов рассматриваемой полугруппы $Z$, и начальное состояние $m^{\mathcal{Z}}_0$ – нейтральный элемент $\mathcal{Z}$, множество функций, меняющих состояние $H_{\mathcal{Z}} = \{h_z \colon h_z(b) = b \bigoplus z \ \forall b \in Z\}_{z \in Z}$ и функция выдачи ответа $\sigma_{id}(z) = z$ для любых состояний $z \in Z$.

Пусть $U$ – некоторый ВИГ над базовым множеством $\mathcal{F}$. Пусть $I = \left\langle X_{int2^*}, V, Z, \rho_{int2^*}, \xi^{\mathcal{Z}}_{int2^*}\right\rangle \in S^{\mathcal{Z}}_{int2^*}$ – вычислительная задача информационного поиска ($V \subset Y^{\mathcal{Z}}_{int2^*}$). Пусть $\mathcal{U}(I, \mathcal{F})$ – множество ВИГ над базовым множеством $\mathcal{F}$, решающих задачу $I$. Сложностью задачи $I$ при базовом множестве $\mathcal{F}$ и объёме $q$ назовём число

$$T(I, \mathcal{F}, q) = \min\{T(U) \colon U \in \mathcal{U}(I, \mathcal{F}) \text{ и } Q(U) \le q\}.$$

Здесь и далее под $T(U)$ будем понимать сложность графа $U$ в худшем случае.

Если $k$ – натуральное число, $S = \langle X, Y, Z, \rho, \xi\rangle$ – тип вычисли-

тельных задач информационного поиска, то обозначим

$$\mathcal{I}(k, S) = \{I = \langle X, V, Z, \rho, \xi\rangle \in S\colon |V| = k\}.$$

Будем исследовать функцию, характеризующую зависимость сложности задач класса ВЗИП $\mathcal{I}(k, S^{\mathcal{Z}}_{int2^*})$ от объёма памяти, доступной для алгоритма решения:

$$\mathcal{T}(k, S^{\mathcal{Z}}_{int2^*}, \mathcal{F}^{\mathcal{Z}}_{int2^*}, q) = \max_{I\in\mathcal{I}(k, S^{\mathcal{Z}}_{int2^*})} T(I, \mathcal{F}^{\mathcal{Z}}_{int2^*}, q).$$

**Теорема.** *Существуют такие положительные константы $c_1, c_2$, что для любой коммутативной полугруппы с нулём $\mathcal{Z} = (Z, \bigoplus)$ и любых натуральных $k \geq 3$ и $2 \leq s < k$ имеет место*

$$\mathcal{T}\left(k, S^{\mathcal{Z}}_{dom2}, \mathcal{F}^{\mathcal{Z}}_{dom2}, c_1\frac{k\log_2 k}{\log_2 s}\right) \leq c_2\frac{s}{\log_2 s}\log_2 k.$$

Доказательство данного результата мы не будем приводить здесь полностью. Опишем лишь основные идеи, используемые в нём. По сути задача состоит в том, чтобы для некоторых констант $c_1, c_2$ получить семейство алгоритмов с указанными выше ограничениями на требуемую память и время работы в худшем случае. Для этого предлагается использовать некоторые идеи из [2]. Во-первых для конкретной задачи с фиксированной базой данных $V$ (положим $k = |V|$) и выбранного параметра $s$, такого что $2 \leq s < k$, необходимо построить дерево подбаз $T^s_V$, каждой вершине которого соответствует некоторая подбаза $V$. Корню дерева соответствует всё $V$, а затем из каждой вершины, которой в соответствие поставлено множество $V'$ с $|V'| > s$, то делим $V'$ на $s$ одинаковых частей $V'_1, \ldots, V'_s$ (если $|V'|$ не делится нацело на $s$, то некоторые части будут содержать на один элемент больше оставшихся). Разбиение следует делать таким образом, чтобы первые координаты записей из множеств с меньшими номерами не превосходили первых координат записей множеств с большими номерами. Затем из $V'$ выпускается $s$ потомков, которым ставятся в соответствие множества $V'_1, \ldots, V'_s$.

По полученному дереву $T^s_V$ строится три как бы параллельные структуры, такие что для каждой вершины $v$ дерева $T^s_V$ каждая из

структур содержит по соответствующей вершине. Пусть это будут вершины $v_1, v_2, v_3$ и вершине $v$ соответствует подбаза $V' \subseteq V$. Тогда прохождение запроса $x = (x'_1, x''_1, x_2)$ в вершину $v_1$ логически означает поиск в базе $V'$ и при этом про все элементы $V'$ известно, что их первая координата не меньше $x'_1$; прохождение $x$ в $v_2$ логически означает поиск в базе $V'$ и при этом про все элементы $V'$ известно, что их первая координата не превосходит $x''_1$; наконец, прохождение запроса $x$ в $v_3$ логически означает поиск в базе $V'$ без каких-либо дополнительных знаний о первых координатах записей из $V'$. Структуры строятся таким образом, что любой запрос идет сначала по одному пути из вершин третьей структуры, а затем разделяется в общем случае и идет по одному пути в первой и по одному пути во второй структуре.

При таком подходе оказывается возможным воспользоваться техникой частичного каскадирования, что позволяет на каждом шаге в определённом смысле запоминать положение запроса по второй координате относительно того множества записей, которое рассматривается в конкретной части создаваемой структуры.

Автор выражает благодарность научному руководителю профессору Э. Э. Гасанову.

**Литература**

1. McCreight E. M. Priority search trees. *SIAM Journal of Computing* 14(2), cc.257-276, 1985.

2. Пивоваров А. П. Функциональная сложность задачи подсчёта для двумерной задачи о доминировании. *Интеллектуальные системы*, Т. 15.

3. Пивоваров А. П. Моделирование вычислительных задач информационного поиска. *Интеллектуальные системы*, Т. 14, вып. 1-4, сс. 229-250, 2010.

4. Chazelle B., Guibas L. J. Fractional cascading: I. A Data Structuring Technique. *Algorithmica* 1, сс. 133-162, 1986.

## МОДЕЛИРОВАНИЕ ДИНАМИЧЕСКИХ БАЗ ДАННЫХ

**Плетнев А.А. (МГУ им. М. В. Ломоносова)**
*PletnevOrel@rambler.ru*

Функционирование базы данных — это обработка потока запросов типа поиск, вставка и удаление. При этом в результате запросов типа вставка и удаление база данных изменяется, а на запросы типа поиск выдается ответ. Если поток запросов на поиск существенно преобладает над запросами на изменение базы данных, то такие базы данных называются статическими. Для исследования таких баз данных предназначены информационные графы [1]. Если же поток запросов на изменение базы данных сравним с потоком запросов на поиск, то такие базы данных называются динамическими, и моделированию таких баз данных посвящена данная работа.

Предлагаемая модель динамических баз данных построена на взаимодействии конечного детерминированного автомата и информационного графа. Задача автомата перестраивать информационный граф при изменении базы данных, тем самым обрабатывая динамические запросы пользователя. Эту структуру будем называть динамическим информационным графом.

В формальном определении понятия ИГ используются 4 множества: множество запросов $X$; множество записей $Y$; *множество $F$ одноместных предикатов*, заданных на множестве $X$; *множество $G$ одноместных переключателей*, заданных на множестве $X$ ( *переключатели* — это функции, область значений которых является начальным отрезком натурального ряда). Понятие *информационного графа* (ИГ) определяется следующим образом. Берется конечная многополюсная ориентированная сеть. В ней выбирается некоторый полюс, который называется корнем. Остальные полюса называются листьями и им приписываются записи из $Y$, причем разным листьям могут быть приписаны одинаковые записи. Некоторые вершины сети (в том числе могут быть и полюса) называются переключательными и им приписываются переключатели из $G$. Ребра, исходящие из каждой из переключательной вершин, нумеруются подряд, начиная с 1, и называются переключательными ребрами. Ребра, не являющиеся переключательными, называются предикатными и им приписываются предикаты из множества $F$. Таким образом нагруженную многополюсную ориентированную сеть называем ИГ над базовым

множеством $\mathcal{F} = \langle F, G \rangle$, где $F = \{f_j, j \in I\}$, $G = \{g_i, i \in J\}$, $I, J$ — конечные множества индексов. Определим три множества функций изменения индексов: $R^1 = \{r_c^1 : (I \cup J \cup Y)^{n_c} \times Y \to I,\ c \in C_1\}$, $R^2 = \{r_c^2 : (I \cup J \cup Y)^{n_c} \times Y \to J,\ c \in C_2\}$, $R^3 = \{r_c^3 : Y^{n_c} \to Y,\ c \in C_3\}$. Введем три множества переменных: $Z = \{z_i\}_{i=1}^{\infty}$, где $z_i$ принимают значения из $I$, $V = \{v_j\}_{j=1}^{\infty}$, где $v_j$ принимают значения из $J$, $W = \{w_k\}_{k=1}^{\infty}$, где $w_k$ принимают значения из $Y$.

Рассмотрим произвольный ИГ $U$. Заменим каждый индекс предиката на некоторую переменную из $Z$, каждый индекс переключателя — на переменную из $V$, а каждую запись — на переменную из $W$. После этого сопоставления получим нагруженный граф, который назовем простым шаблоном. Если каждый индекс предиката мы заменим на некоторую формулу над множеством переменных $Z$ и множеством функций $R^1$, каждый индекс переключателя — на формулу над множеством переменных $V$ и множеством функций $R^2$, а каждую запись — на формулу над множеством переменных $W$ и множеством функций $R^3$, то полученный нагруженный граф назовем шаблоном.

Будем говорить, что ИГ $U$ и простой шаблон $\mathcal{T}$ согласованы, если они совпадают как графы, и если в ИГ $U$ встречаются одинаковые индексы предикатов (переключателей и записей), то в соответствующих местах шаблона $\mathcal{T}$ находятся одинаковые переменные из $Z$($V$ и $W$). Возникшее соответствие между переменными и индексами назовем интерпретацией данного согласования.

Локальным преобразованием назовем пару $p = (\mathcal{T}_1, \mathcal{T}_2)$, где $\mathcal{T}_1$ — простой шаблон, $\mathcal{T}_2$ — шаблон, в формулах которого встречаются только переменные, входящие в простой шаблон $\mathcal{T}_1$, и возможно еще одна переменная из множества $W$.

Если ИГ $U$ и простой шаблон $\mathcal{T}_1$ согласованы, то применением локального преобразования $p = (\mathcal{T}_1, \mathcal{T}_2)$ к ИГ $U$ назовем ИГ $U'$, получающийся из шаблона $\mathcal{T}_2$ подстановкой вместо каждой формулы значения данной формулы в интерпретации согласования ИГ $U$ и простого шаблона $\mathcal{T}_1$.

Пример преобразования $p$ показан на Рис. 1, где $r^2(v_1, v_2, w) \in R^2; r_1^3(w_1, w_2, w), r_2^3(w_1, w_2, w), r_3^3(w_1, w_2, w) \in R^3$, $a = (w_1, w_2, w)$, $b = (v_1, v_2, w)$, $c = \min(v_1, v_2, w)$, $r^2(b) = (c, \min(\{v_1, v_2, w\} \backslash \{c\}))$, $r_1^3(a) = \min(w_1, w_2, w)$, $r_3^3(a) = \max(w_1, w_2, w)$, $r_2^3(a) = \{w_1, w_2, w\} \backslash \{r_1^3(a), r_3^3(a)\}$.

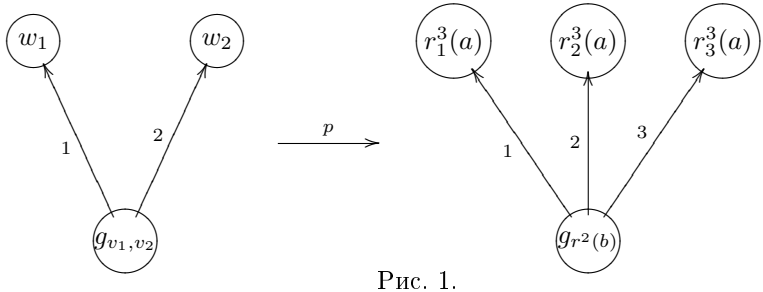



Рис. 1.

Такое преобразование используется при добавлении новой записи в 2–3 деревьях.

Далее будем рассматривать класс информационных графов $\mathcal{K}(N)$, $N \in \mathbb{N}$, таких, что степень инцидентности любой вершины графа не превосходит $N$. Окрестностью радиуса $L$ некоторой вершины ИГ, будем называть множество вершин ИГ, таких что длина пути до которых от данной не превосходит $L$, и множество ребер, соединяющих эти вершины, вместе с нагрузками этих вершин и ребер.

Назовем кодом вершины ИГ пару $(k_1, k_2)$, где $k_1 = 0$, если вершина предикатная; $k_1 = 1$, если она переключательная; $k_2 = 0$, если вершина корень; $k_2 = 1$, если вершина листовая; $k_2 = 2$ в остальных случаях. Кодом ребра назовем число, которое равняется 0, если оно предикатное и 1, если переключательное.

Кодом окрестности на запросе $x$ назовем информацию о коде каждой вершины и ребра данной окрестности, а также информацию о значениях всех предикатов и переключателей на запросе $x$ (так же возможно, что код содержит дополнительную информация о пометках на ребрах и вершинах графа).

Пусть $\mathcal{P}$ — некоторое конечное множество локальных преобразований, $L, N$ — натуральные числа, $U$ — ИГ над базовым множеством $\mathcal{F} = \langle F, G \rangle$ из класса $\mathcal{K}(N)$, $\mathcal{A}$ – конечный автомат, входной алфавит, которого есть множество кодов всевозможных окрестностей радиуса $L$ вершин ИГ из , а выходной алфавит описывает реакции автомата, такие как перемещение автомата по графу, выбор локального преобразования из $\mathcal{P}$, сигнал на завершение работы и, возможно, расстановку или изменение пометок на рассматриваемой окрестности графа. Пару $(\mathcal{A}, U)$ назовем динамическим информационным графом (ДИГ) над $\mathcal{F}$ и $\mathcal{P}$.

Определим *функционирование* ДИГ $(\mathcal{A}, U)$ на запросе. Если запрос есть запрос на поиск, то функционирование ДИГ совпадает с функционированием ИГ $U$ и не задействует автомат $\mathcal{A}$. Если запрос является запросом на вставку или удаление, то функционирование ДИГ происходит следующим образом. В начальный момент текущей вершиной объявляется корень ИГ и считается, что лишних пометок на графе нет. На вход автомата подается код окрестности текущей вершины. Если выход автомата предписывает передвижение по графу, то текущая вершина изменяется. Если выход автомата предписывает некоторое локальное преобразование, то в случае если окрестность текущей вершины согласована с левой частью преобразования, то рассматриваемая окрестность заменяется на результат применения данного локального преобразования. Если окрестность текущей вершины не согласована с левой частью преобразования, то функционирование завершается с ошибкой. Если выход автомата предписывает изменение пометок рассматриваемой окрестности, то эти изменения выполняются. Если выход автомата сигнализирует о завершении работы, то обработка запроса завершается успешно.

Пусть дана задача информационного поиска (ЗИП) $I = \langle X, V, \rho \rangle$. Скажем, что ДИГ решает ЗИП $I$, если ответ на произвольный запрос $x$ типа поиска, содержит те и только те записи $y \in V$, что $x \rho y$; если функционирование на произвольном запросе типа вставки (удаления) записи $y \in Y$ завершается успешно, результирующий граф не содержит лишних пометок и решает ЗИП $\langle X, V \cup \{y\}, \rho \rangle$ ($\langle X, V \setminus \{y\}, \rho \rangle$).

Если каждому локальному действию сопоставить число, характеризующее его сложность, то можно вводить сложность ДИГ на запросах, и другие сложностные характеристики ДИГ.

Далее можно ставить задачу синтеза, а именно задачу построения для произвольной ЗИП такого ДИГ, который бы решал данную ЗИП и имел при этом как можно меньшую сложность.

Автор благодарит профессора Э. Э. Гасанова за постановку задачи и помощь в работе.

**Литература**

1. Э. Э. Гасанов, В. Б. Кудрявцев. Теория хранения и поиска информации. М.: ФИЗМАТЛИТ, 2002.

## КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ЛОГИЧЕСКИХ ПРОЦЕССОВ

**Подколзин А.С. (Московский государственный университет им. М.В.Ломоносова)**

Мечта человечества об искусственном интеллекте имеет давнюю историю. На сегодняшний день она, увы, остается лишь мечтой. Производительность и память современных вычислительных систем огромны и продолжают быстро увеличиваться. Они относительно дешевы и широко распространены. Однако, среднестатистический стандарт использования этих систем по-прежнему сводится к некоему гибриду телефона, записной книжки, пишущей машинки, телевизора и тренажера для отработки простейших реакций. Понимание мира хотя бы на уровне пятиклассника, с его способностью читать книги и воспринимать смысл изображений, остается далеко за рамками возможностей существующих "интеллектуальных систем". О таких вещах, как научно-техническое творчество в подлинном смысле этого слова, говорить и вовсе не приходится. Автоматизировать удается лишь сравнительно простые и узко специализированные функции естественного интеллекта. Так как данное положение наблюдается уже достаточно давно, есть все основания говорить о наличии некоторого принципиального барьера, возникшего на пути к искусственному интеллекту. Очевидно, этот барьер никак не связан с "физическими" возможностями сегодняшних компьютеров, а связан с острым дефицитом наших знаний о логических процессах, лежащих в основе любой интеллектуальной деятельности. По существу, речь должна идти о создании новой науки, изучающей эти процессы, своего рода "логической динамики".

Нельзя сказать, чтобы попыток начать изучение логических процессов не было вовсе. Прежде всего, они предпринимались в рамках математической логики. Однако, ее попытки создания универсальных эффективных процедур решения задач успехом не увенчались. Из общих соображений возникла лишь общая схема организации перебора. Экспоненциально растущая трудоемкость ограничила сферу применимости этих процедур самыми простыми задачами и даже поставила под сомнение возможность создания искусственного интеллекта вообще. Впрочем, человек вполне эффективно решает задачи, избегая "экспоненциальных переборов и неудача попытки решить

проблему математическим путем означает лишь заведомую недостаточность умозрительных построений для анализа такого сложного явления, как интеллект.

С другой стороны, практически сразу после появления компьютеров стали создаваться программы для решения "интеллектуальных" задач в самых различных областях. Они имитировали действия эксперта в соответствующей области и получили название экспертных систем. Рассматривались задачи на доказательство теорем в формальной логике, задачи на доказательство геометрических теорем, формальное интегрирование, тригонометрические уравнения, уравнения в целых числах, шахматные задачи, и многое, многое другое. Собственно, сейчас трудно найти такую предметную область, для которой не было бы создано множество экспертных систем. Некоторые из них весьма развиты и успешно могут конкурировать с человеком. Например, для шахмат была создана система, одержавшая победу даже над чемпионом мира. Может быть, некое объединение всех этих систем и является долгожданным искусственным интеллектом? Однако, при ближайшем рассмотрении становится ясно, что богатство здесь иллюзорное. В сколь-нибудь сложных областях развитие экспертной системы останавливалось на простейших задачах, демонстрируя скорее невозможность освоения области современными средствами, чем успешное ее преодоление. В отдельных случаях (в тех же шахматах) удавалось достичь внешне впечатляющих результатов не за счет проникновения в логику действий человека, а лишь методом "грубой силы используя огромную производительность компьютеров для перебора. Хорошо известны высказывания на этот счет одного из пионеров в области искусственного интеллекта чемпиона мира по шахматам М.М.Ботвинника.

Разумеется, во многих, сравнительно простых с логической точки зрения, предметных областях были созданы эффективные прикладные экспертные системы. Однако, здесь возникло другое негативное явление. Разработчики, находясь под впечатлением достигнутых успехов, экстраполировали принципы организации своих систем далеко за рамки рассматривавшейся предметной области, возводя их в ранг "общей теории" искусственного интеллекта. Естественно, в сколь-нибудь более сложных или просто сильно отличающихся от исходной предметных областях эти теории никакой пользы не принесли. Здесь прослеживается всё та же линия основанных на недо-

статочной информации умозрительных построений.

Впрочем, отвлечемся от заведомой слабости какой-то (неважно, малой или большой) части узкоспециализированных экспертных систем. Предположим, что каждая из них в своей области достигла уровня самого опытного эксперта. И в этом случае формальное их объединение никакого искусственного интеллекта не даст. Во-первых, из-за проблем организации взаимодействия. Как известно, знания различных предметных областей тесно взаимосвязаны друг с другом. Нельзя хорошо решать задачи по геометрии, совсем не зная алгебры, нельзя овладеть физикой, не зная математики, и т.д. и т.п. Чтобы научить систему пониманию смысла изображений, нужно заложить в нее примерно столько же знаний о мире и приемов использования этих знаний в рассуждениях, сколько их понадобилось бы для обучения другой системы пониманию естественного языка. Никакой самый умный переключатель между изолированными экспертными системами здесь не поможет, так как решение задач "на стыке" различных областей потребует одновременного их участия. Если бы подобная попытка создания "объединенного" искусственного интеллекта и в самом деле когда-либо была бы предпринята, то она немедленно привела бы к необходимости устранения перегородок между отдельными частями и, фактически, созданию заново некоей "универсальной" экспертной системы.

Но и так мы не получили бы искусственного интеллекта. Экспертная система лишь зафиксировала бы текущий уровень развития наших знаний и умений. Несмотря на свою неоспоримую практическую полезность, она никоим образом не могла бы претендовать на ту роль мощного ускорителя научно-технического прогресса, которую призван сыграть искусственный интеллект. Такая роль однозначно предполагает способность системы к саморазвитию. Простейшая адаптация, связанная с оптимизацией системой каких-то своих параметров, не в счет. Подлинное саморазвитие требует механизмов, создающих новые приемы решения задач на основе имеющихся знаний и пополняющих знания с ориентацией на решение задач. Без этого невозможно решение нестандартных задач, а они в творчестве составляют подавляющую долю. Интеллектуальной системе просто необходима мощная техника компиляции, выводящая ее архитектуру и программы из идей, порождаемых ею же на языке, максимально приближенном к языку теоретических знаний. Собственно гене-

рация идей происходит на пограничном слое между теоретическими знаниями и практическими приемами, который и должен стать главным объектом изучения.

Здесь мы снова приходим к вопросу о целесообразности создания изолированных друг от друга интеллектуальных систем в различных предметных областях. Конечно, для игры в шахматы, знание, скажем, аналитической геометрии может показаться излишним. Однако, механизмы, порождающие новые приемы на основе теоретических знаний, имеют общелогический характер. Каждая предметная область вносит в копилку таких механизмов, типов приемов и стандартов рассуждений что-то свое. И для получения полной коллекции необходимо изучить как можно более широкий спектр различных предметных областей, чтобы уже впоследствии вернуться "во всеоружии" к какой-то одной области.

Таким образом, путь к искусственному интеллекту лежит через создание экспертной системы, охватывающей достаточно широкий спектр предметных областей и программируемой на уровне "пограничного слоя" между теорией и алгоритмами. Она должна послужить своего рода микроскопом, дающим достаточно богатый фактический материал для изучения общих принципов организации логических процессов: принципов эффективного управления рассуждениями при решении задач, принципов извлечения новых приемов из теорем, принципов автоматического развития теорий, и т.д. Альтернативой является лишь продолжение бесплодных попыток умозрительного угадывания этих принципов.

Данная система вовсе не обязана претендовать на роль сколь-нибудь сильного решателя. В первую очередь, она должна давать множество примеров, объясняющих, как можно было бы управлять рассуждениями, чтобы решить ту или иную конкретную задачу, не прибегая к непомерно большому перебору. Разумеется, по мере накопления средств она начнет многое делать самостоятельно. Однако, в иных предметных областях возможность полной проработки темы при обучении "вручную" вообще представляется сомнительной, и здесь придется ограничиться накоплением единичных траекторий процессов, которые все же дадут пищу для обобщений. Действительно сильные решатели должны будут появиться впоследствии, когда накопление знаний о логических процессах позволит создать интеллектуальные системы, способные самообучаться. Пока таких систем

нет, и речь идет лишь о начале систематического исследования, опирающегося на некий компьютерный "логический микроскоп".

Предлагаемая работа посвящена развитию компьютерной системы, которая могла бы стать основой для универсальной экспертной системы ("решателя") указанного выше типа и одновременно "микроскопом" для изучения общих принципов организации логических процессов, включая самообучение. Система основана на применении нового языка программирования ГЕНОЛО́Г, расположенного в точности на пограничном слое между теорией и алгоритмами. Прием на этом языке задается как теорема предметной области, снабженная некоторой алгоритмизирующей разметкой.

Обучение системы предпринималось в широком спектре предметных областей и позволило выявить достаточно богатую коллекцию способов эффективной алгоритмизации теорем. Использование этой коллекции, с одной стороны, существенно ускорило и упростило процесс "ручного" обучения решателя, а с другой стороны - позволило вплотную приблизиться к автоматическому синтезу приемов.

Следует заметить, что не только программирование решателей, но даже традиционное "нелогическое" программирование в конечном счете выводит свои конструкции из теорем. Это означает, что в самообучающихся интеллектуальных системах вообще все программирование неизбежно должно будет проходить через уровень ГЕНОЛО́Га, и он вполне может претендовать на роль "универсального алгоритмического языка будущего". Разумеется, это придает особую значимость развитию компиляторов для языков такого типа.

При развитии ГЕНОЛО́Га и обучении решателя, происходивших одновременно, были рассмотрены такие предметные области, как элементы дискретной математики, алгебра множеств, элементарная алгебра, элементарная геометрия, аналитическая геометрия, линейная алгебра, математический анализ, дифференциальные уравнения, комплексный анализ, вычисления, теория вероятностей, элементы общей алгебры, ряд разделов элементарных физики и химии, шахматы, текстовые задачи (в двух вариантах - на логическом либо естественном языке), анализ рисунков. Обучающий материал содержал примерно 11000 задач, из которых были извлечены примерно 39000 приемов. В отдельных областях даже такое, сугубо предварительное, обучение позволило выйти на неплохой уровень решения задач средней сложности. В других - удалось проработать

лишь небольшое количество примеров, которые, однако, дали возможность существенно скорректировать первоначальные представления и получить сравнительно устойчивую архитектуру для дальнейшего развития.

Предпринимается анализ возможностей самообучения системы. К явлению самообучения следует относиться с определенной осторожностью, так как пока известна лишь одна самообучающаяся в абсолютном смысле интеллектуальная система - это вся человеческая цивилизация в целом. Процессы подобного масштаба далеко выходят за рамки возможностей современных компьютеров. Практический интерес представляет сейчас "ограниченное" самообучение отдельного человека, заключающееся в самостоятельном усвоении по книгам или иным источникам возможно большей суммы знаний и приобретении навыков их практического использования путем тренировки на обучающих примерах. Целью работы с компьютерной системой должно явиться освоение аналогичного режима самообучения: ей сообщается теория и дается поток задач, на котором происходят самостоятельное создание приемов и оптимизация их совместного поведения.

Центральным здесь является создание развитой классификации логических типов приемов, которая позволяла бы по заданной теореме генерировать серию предположительно интересных для задачи приемов ("идей"), проверять их ценность и отбирать по окончании решения задачи те приемы, которые оказались результативными. Работа над созданием такой классификации начата, однако она связана с трудоемкой итеративной стандартизацией приемов созданного (в общем, пока относительно "сырого") решателя и оптимизацией его логических режимов.

Архитектура и языки обучения системы описаны в первом томе монографии по компьютерному моделированию логических процессов [1]. Планируется издание последующих томов, в которых будут описаны многообразие приемов решателя, возникших при его обучении, и средства автоматизации синтеза приемов.

Автор выражает искреннюю благодарность В.Б.Кудрявцеву, поддержка которого сделала возможным проведение данного исследования.

**Литература**

1. Подколзин А.С. Компьютерное моделирование логических процессов. Архитектура и языки решателя задач. М., Физматлит, 2008.

# О НЕЛИНЕЙНЫХ ХАРАКТЕРИСТИКАХ НЕЙРОННЫХ СХЕМ В ПРОИЗВОЛЬНЫХ БАЗИСАХ

**Половников В.С. (Московский Государственный Университет им. М.В.Ломоносова)**
*pvser@mail.ru*

В данной работе получены следствия некоторых результатов моей кандидатской диссертации, которые могли бы быть полезны при построении нейронных схем из элементов, представляющих собой полную (по операции суперпозиции) систему $F$ кусочно-параллельных функций, содержащих все линейные элементы (сложение, константы из $\mathbb{R}$, умножения на константы из $\mathbb{R}$) и некоторую нелинейную часть.

Нам понадобятся слующие определения, обозначения и утверждения диссертации (нумерация теорем сохранена):
$L$ — множество линейных функций,
$PC$ — множество кусочно-постоянных функций,
$PP = \{f | f = f_c + f_l, f_c \in PC, f_l \in L\}$ — множество кусочно-параллельных функций.
Нейронной схемой Мак-Каллока-Питтса называется схема, построенная с использованием линейных элементов и нелинейной $\theta$-функции Хевисайда.

$$\theta(x) = \begin{cases} 1, & \text{если } x \geq 0 \\ 0, & \text{если } x < 0. \end{cases}$$

**Теорема 3.** *Множество функций, реализуемых нейронными схемами Мак-Каллока-Питтса, совпадает с множеством всех кусочно-параллельных функций. И для любой кусочно-параллельной функции существует нейронная схема Мак-Каллока-Питтса нелинейной глубины не более двух.*

Функция $f(t)$ из $PC(1)$ называется *С-финитной функцией от* 1 *переменной*, если $\exists T_f, c_f \in \mathbb{R}$, $T_f > 0$ такие, что при $t \in (-\infty, -T_f) \cup (T_f, +\infty)$ выполнено $f(t) = c_f$. Обозначим класс С-финитных функций от 1 переменной $\Phi(1)$.

Рассмотрим кусочно-постоянную функцию $f : \mathbb{R}^n \to \mathbb{R}$ и прямую $\mathfrak{N}$ заданную параметрически: $x_1 = a_1 t + b_1, \ldots, x_n = a_n t + b_n$. Функция $g(t) = f(a_1 t + b_1, \ldots, a_n t + b)$ называется *сечением кусочно-постоянной функции $f$ прямой $\mathfrak{N}$*.

Функция $f \in PC$ называется *C-финитной*, если сечение $f$ любой прямой является C-финитной функцией от 1 переменной. Обозначим класс всех C-финитных функций через $\Phi$.

Введем *класс C-финитно-линейных функций*:

$$\Phi L = \{f | f = \phi + l, \phi \in \Phi, l \in L\}.$$

**Теорема 4.** *$F$ полно в $PP$ по операции суперпозиции тогда и только тогда, когда $F \not\subseteq \Phi L$.*

Пусть $f(x_1, \ldots, x_n)$ — произвольная кусочно-параллельнейная функция в $\mathbb{R}^n$, заданная $k$ гиперплоскостями. За $Z_{MP}(S)$ обозначим число элементов $\theta$ в схеме $(S, f)$, то есть *нелинейную сложность* схемы $(S, f)$. Соответственно определим *сложность реализации функции* схемой Мак-Каллока-Питтса

$$Z_{MP}(f) = \min_{S \text{ реализует} f} Z_{MP}(S),$$

где минимум берется по всем нейронным схемам Мак-Каллока-Питтса реализующим $f$.

*Функция Шеннона* $Z_{MP}(k) = \max\limits_{f \in PP: f \text{ задана } k \text{ гиперплоскостями}} Z_{MP}(f)$.

**Теорема 5.** *Для функции Шеннона $Z_{MP}(k)$ верна оценка*
*1) при $k \leq n$:*

$$3^k - 1 \leq Z_{MP}(k) \leq 3^k + 2k.$$

*2) при $k > n$:*

$$3\sum_{i=0}^{n-1} C_{k-1}^i 2^i + 2^n \sum_{j=n-1}^{k-2} C_j^{n-1} - 1 \leq Z_{MP}(k) \leq 3\sum_{i=0}^{n-1} C_{k-1}^i 2^i + 2^n \sum_{j=n-1}^{k-2} C_j^{n-1} + 2k.$$

Из докозательства теоремы 4 вытекает выразимость $\theta$ функции через нефинитно-линейную функцию из $F$, которая согласно теореме

4 обязана содержаться в $F$ ввиду полноты $F$. Причем схема строится явно со следующими параметрами: нелинейная глубина 4 и нелинейная сложность 4. Для произвольной кусочно-параллельной функции, рассмотрим ее нейронную схему Мак-Каллока-Питтса нелинейной глубины 2 (из теоремы 3). Заменим в ней элемент $\theta$ на нейронную схему над $F$ (из теоремы 4). Получаем следущее утверждение:

**Следствие 1.***Для любой кусочно-параллельной функции существует нейронная схема над $F$ нелинейной глубины не более* 8.

Далее, аналогично $Z_{MP}(k)$ определим функцию Шеннона $Z_F(k)$ для функций, реализуемых схемами над $F$. Аналогичной заменой элемента $\theta$ на нейронную схему над $F$, обобщается верхняя оценка функции Шенона из Теоремы 5.

**Следствие 2.***Для функции Шеннона $Z_F(k)$ верна оценка 1) при $k \leq n$:*

$$Z_{MP}(k) \leq 4(3^k + 2k).$$

*2) при $k > n$:*

$$Z_{MP}(k) \leq 4(3\sum_{i=0}^{n-1} C_{k-1}^i 2^i + 2^n \sum_{j=n-1}^{k-2} C_j^{n-1} + 2k).$$

За помощь в научных изысканиях выражаю благодарность к.ф.-м.н. Часовских А.А.

## Литература

1. Половников В. С. Об оптимизации структурной реализации нейронных сетей. Дис. –М., 2007

2. Автоматы. Сборник статей. Под ред. К. Э. Шеннона и Дж. Маккарти. Пер. с англ. под ред. А. А. Ляпунова. –М., Изд. иностр. лит., 1956.

3. Хайкин С. Нейронные сети: полный курс, 2-е издание, –Вильямс, 2006. 4. McCulloch W. S. and W. Pitts A logical calculus of the ideas immanent in nervous activity –Bulletin of Mathematical Biophysics, vol. 5, p. 115-133, 1943.

5. Кудрявцев В. Б. Функциональные системы. –М., Изд-во МГУ, 1982.

6. Лупанов О. Б. Асимптотические оценки сложности управляющих

систем. –М., МГУ, 1984.
7. Кудрявцев В. Б., Алешин С. В.,Подколзин А. С. Введение в теорию автоматов. –М.:Наука.Гл.ред.физ.-мат.лит., 1985.
8. F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization of the brain –Psychological Review, 65:386-408, 1958.
9. Кофман А. Введение в прикладную комбинаторику. –М.: Наука, 1975.
10. Гельфонд А. О. Трансцендентные и алгебраические числа. Изд.2 –М.: КомКнига, 2006.

# МОДЕЛИРОВАНИЕ ЭЛЕКТРОННОГО БИЛИНГВАЛЬНОГО МЕТОДИЧЕСКОГО СЛОВАРЯ ОТКРЫТОГО ТИПА КАК СОСТАВЛЯЮЩАЯ ПРОЦЕССА ФОРМИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ

**Романова А.А. (Смоленск)**
*Vackhanka88@mail.ru*

Одной из задач, решаемых интеллектуальными информационными системами, является обучение. Это подразумевает использование компьютера для обучения какой-либо дисциплине или предмету. Системы обучения диагностируют ошибки при изучении дисциплины с помощью ЭВМ и подсказывают правильные решения. Структура интеллектуальной системы включает три основных блока - базу знаний, решатель и интеллектуальный интерфейс[2]. Таким образом, рассматривая электронный билингвальный словарь открытого типа как базу знаний, его можно отнести к одному из составляющих компонентов интеллектуальных информационных систем, поскольку в данном случае высококвалифицированными специалистами, осуществляющими выявление, исследование и применение знаний, являются одновременно педагогики, учителя-предметники и лингвисты. При построении интеллектуальных систем, основанных на знаниях, используются знания, накопленные экспертами в виде конкретных правил решения тех или иных задач, следовательно, билингвальный методический словарь-справочник, созданный по принципу Википедии, является типичным банком знаний, который впоследствии может быть включен в состав более сложной экспертной системы. Являясь составной частью интеллектуальной информационной системы, такой веб-ресурс может также быть классифицирован как гипертекстовая система.

В свою очередь билингвальное образование в настоящее время становится общезначимым национальным предписанием, причем не только нравственного, но и профессионального характера. Отсюда возникает актуальность проблемы повышения квалификации преподавателя высшей профессиональной школы в системе непрерывного (lifelong) и рекуррентного, возобновляемого (recurrent) образования. Становится очевидной необходимость разработки двуязычных электронных методических справочников с учетом отдельных специальностей методистов и педагогов, поскольку обучение на билинг-

вальной основе способствует углублению предметной подготовки и расширению сферы межкультурного обучения, совершенствованию общей языковой подготовки и владения иностранным языком в специальных неязыковых предметных целях, а также повышению мотивации в изучении иностранного языка[3].

Одним из главных этапов в разработке любой интеллектуальной информационной системы, в том числе и гипертекстовой, является построение концептуальной модели. В рамках нашего исследования было создано целостное и системное описание используемых знаний, отражающее сущность функционирования проблемной области. В результате концептуализации проблемы проектирования электронного методического словаря открытого типа была получена объектная модель, описывающая структуру предметной области как совокупности взаимосвязанных объектов (Схема 1).

Процесс проектирования электронного билингвального методического словаря в данной концептуальной модели представляет собой алгоритм действий, после выполнения которых мы достигнем поставленной цели. Первый шаг алгоритма - постановка проблемы - включает в себя определение типа проблемы, цели и задач [2]. Целью концептуальной модели является повышение уровня знаний и профессиональной языковой компетенции учителей Вузов и педагогов-исследователей.

Следующий этап - это параллельное выполнение двух шагов алгоритма: определение теоретико-методологических основ процесса проектирования, а также выявление и анализ потребностей конечных пользователей к предполагаемому продукту. Параллельное выполнение этих действий обеспечивает их взаимовлияние и взаимообусловленность. При проектировании концептуальной модели нами были взяты за основу следующие методологические подходы: системный, билингвальный, культурологический, рефлексивный, деятельностный. Системный, билингвальный и культурологический подходы позволяют выявить сущностные характеристики, закономерности и принципы интерсоциальной концепции развития межкультурной и языковой профессиональной компетенции студентов, уровни ее функционирования, построить и обосновать концептуальную модель[4]. Рефлексивный и деятельностный подходы обеспечивают разработку методических основ развития профессиональной языковой компетенции студентов в процессе образова-

ния в неязыковом вузе. Данные подходы входят в состав теоретико-методологического компонента концептуальной модели и позволяют положить начало реализации ведущих идей данного исследования - глобализации и модернизации. Актуальность идеи модернизации связана с тем, что полингвальное обучение, являясь средством получения образования, представляет собой также и процесс формирования личности, открытой к взаимодействию с окружающим миром [5]. А идея глобализации ставит цель включения подобного ресурса в состав обучающей экспертной системы, доступной широкой аудитории. Теоретико-методологический компонент концептуальной модели также раскрывает общедидактические принципы реализации указанных подходов: принцип научности, доступности, мотивации и наглядности.

После определения теоретико-методологического компонента и анализа потребительских требований следует этап непосредственного моделирования проекта, пока еще не окончательного оформления идеи, иначе - организационно-методический компонент. Проектирование методического ресурса обуславливает наличие у данного этапа характерных особенностей: методические приемы и условия оформления проекта в некоторую оболочку. Педагогические условия включены в концептуальную модель, так как система не может быть создана и существовать в ином виде, кроме как в комплексе с указанными условиями, иначе мы получим систему с другими характеристиками [4].

Объединяя методические приемы и средства создания, можно выделить 8 этапов в создании электронного билингвального методического словаря открытого типа: 1) анализ литературы, 2) разбиение материала на модули, 3) определение гиперссылок, 4) реализация в электронной форме, 5) разработка компьютерной поддержки, 6) корректировка материала, 7) отбор мультимедийного сопровождения и 8) возможная визуализация материала [1]. Далее алгоритм нашего проектирования предусматривает проведение экспертного анализа с последующей корректировкой в случае необходимости. И только после этого следует практическая апробация нашего методического словаря открытого типа. Практическая часть эксперимента реализуется в соответствии с разработанной учебной программой при условии активной билингвальной учебной деятельности учащихся и направляющей билингвальной педагогической деятельности учите-

ля.

В качестве результата функционирования предложенной системы мы рассматриваем определенный уровень методической подготовки учащихся, который обеспечивает переход системы на новую, более высокую стадию развития. Оценочно-результативный компонент включает уровни методической готовности к педагогической деятельности учащихся, критерии и показатели, диагностику и методы математической статистики.

Таким образом, спроектированная нами модель обладает всеми признаками системы, а именно:

- целостностью, так как все указанные компоненты взаимосвязаны между собой, выполняют определенную функцию, способствуют достижению планируемого результата, который выражается в высоком уровне билингвальной методической готовности студентов;

- наличием инвариантных компонентов (социальный заказ, государственный образовательный стандарт, цель, теоретико-методологические основы решения проблемы) и вариативных компонентов (средства, механизмы достижения цели);

- открытостью, т.к. каждый компонент взаимодействует с внешней средой: данная система, с одной стороны, сама испытывает влияние среды, с другой, - оказывает на неё влияние, организуя её в соответствии с целью;

- динамичностью, т.к. содержание компонентов может меняться в зависимости от социального заказа и предполагает совершенствование в процессуальном плане и в оценке качественной характеристики результата - уровней;

- линейно-возвратным характером, выражающимся в обеспечении оперативной обратной связи, корректирующей недостатки полученного результата [4].

Спроектированная нами модель является концептуальной, т.к. учитывает основные положения интерсоциальной концепции, закономерности и принципы её построения и направлена на их реализацию.

**Литература**

1. Зимина О.В. Печатные и электронные учебные издания в современном высшем образовании: Теория, методика, практика. - Москва, МЭИ, 2003.

2. Гаврилова Т.А. Базы знаний интеллектуальных систем. - СПб, Питер, 2000.

3. Мафтей А.Г. Личностно-ориентированный подход в управлении поликультурной школой: автореферат дисс-и канд.пед.наук. - Смоленск, Приднестровский гос. университет, 2007.

4. Павлова Л.В. Развитие гуманитарной культуры студентов ВУЗа: монография. - Москва, "Академия Естествознания 2010.

5. Салехова Л.Л. Дидактическая модель билингвального обучения математике в высшей педагогической школе: автореферат дисс-и доктора пед.наук.- Казань, 2007.

# ОБЪЕКТНАЯ СУБД DIM И ПУТИ ЕЕ РЕАЛИЗАЦИИ

**Рублев В.С., Смирнова Е.А.**
**(Ярославский госуниверситет им. П.Г. Демидова)**
*roublev@mail.ru*

Требования современного развития баз данных ставят новые задачи. Последние достижения в научных исследованиях (например, в астрофизике) требуют хранения и обработки колоссальных объемов информации, которые подошли уже к пентабайтам [1]. Потребность в постоянных изменениях не только данных, но и алгоритмов является "неизменной характеристикой современного мира" [2].

В [3] описан новый объектный подход к созданию СУБД, который предполагает не только изменение данных объектов, но и возможность изменения типов объектов, т. е. схемы базы данных, названный *динамической информационной моделью*[1] (DIM). В этом подходе мы выделили 6 базовых отношений объектов: *наследования, включения, внутреннего наследования, внутреннего включения, истории и взаимодействия.* Для описания предметной области введено общее определение Дискретной детерминированной модели, объекты которой могут эволюционировать, дана формализация таким моделям (OD-модели) и показана полнота DIM: *любая OD-модель и ее эволюция может быть описана с помощью DIM.* Описанный в [4] язык объектно-динамических запросов ODQL для динамической информационной модели DIM обладает полнотой – *любая группа объектов DIM (вместе с любыми их свойствами) может быть выделена ODQL-запросом* (см. [5]). В [6] описана организация выполнения запросов, при которой трудоемкость оптимальна. Но время выполнения запросов зависит как от организации хранения данных и манипулирования ими, так и от выбора платформы реализации. При этом должны быть учтены вопросы динамического изменения данных. Рассмотрим сначала пути получения такой организации, которые не зависят от выбора платформы для реализации.

Естественный путь организации динамической СУБД может быть определен метауровенем описания объектов, классов объектов, парметров и реляционных связей этих сущностей. Метауровень представляет собой реляционную базу данных с набором таблиц, со-

[1] В алгебре система объектов с введенными отношениями называется *моделью.*

держащих в себе все нужные значения. При этом выбор в качестве платформы реляционной СУБД может быть сделан позже на основе анализа удачности выбора той или иной платформы для метауровня. В таблицах метауровня информация о классах представлена как об объектах метауровня. Поскольку параметры каждого класса могут меняться, то информация о параметрах классов также представлена в других таблицах, связанных с таблицей классов реляционными связями. Объекты каждого класса организуются в отдельных таблицах, связанных с классами таблицы классов. Подобным образом и значения параметров объектов поместим в отдельные таблицы, связанные с соответствующими таблицами параметров и объектов реляционными связями. Отметим, что каждый класс, параметр, объект имеет свой уникальный идентификатор (IdClass, IdParamter, IdObj) для установления упомянутых связей между таблицами, а также 2 атрибута *Дата рождение* и *Дата смерти*, определяющими период жизни соответствующего данного (при изменении класса, параметра, объекта их объектные идентификаторы также изменяются). На рисунке 1 приведена схема организации метауровня DIM. На этой схеме ClassInheritance, ClassInclusion, ClassInteraction, ClassHistory таблицы связей классов, соответствующих отношениям наследования, включения, взаимодействия и иерархии.Таблица Obj_IdClass объектов класса определяет для каждого объекта пометку Mark выбора объекта в *индекс-выборку класса* (см. [6]), а сами индекс-выборки классов и отношений классов реализуются как внешние индексы этой таблицы и таблиц связей объектов.

Введенный в [4] SQL-подобный язык объектных запросов ODQL выделяет из фразы **WHERE** ограничения **FOR**, накладываемые на данные того или иного класса объектов, и ограничения **LINKS**, накладываемые на связи классов в порядке обхода *схемы слоев* (см. [5]). Алгоритм выполнения запроса состоит из этапов начального определения индекс-выборок классов по фразе **FOR**, коррекции этих индекс-выборок с использованием отношений классов в порядке обратном связям во фразе **LINKS** и получения в качестве результата "деревьев" данных фразы **SELECT** для каждого объекта базового класса (первого во фразе **FROM**).

В качестве платформы метауровня предполагается база данных JAVA. В настоящее время проводится сравнительное исследование быстроты выполнения запросов с использованием вышеуказанного

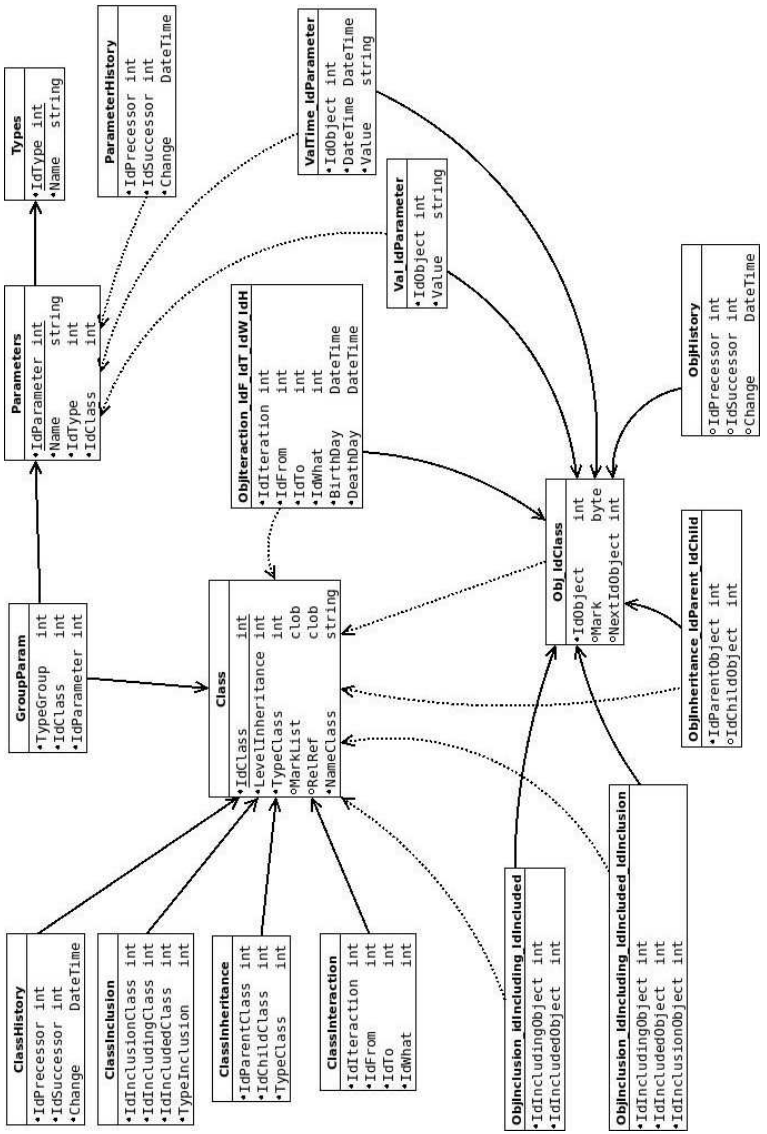



Рис. 1: Схема метауровня

алгоритма и с помощью другого алгоритма обхода схемы слоев, а также сравнение с быстротой выполнения преобразованных ODQL-запросов в последовательности SQL-запросов. Анализ этих исследований должен выявить область использования каждого из подходов, что даст возможность перейти к лучшей реализации DIM.

**Литература**

1. Gray J., Liu D.T., Nieto-Santisteban M., Szalay A., Dewitt D.J., Heber G. Sientific Data Management in the Coming Deacade. SIGMOD Recjrd, V. 34, no 4, December 2005.

2. Сивцов А. Шесть компонентов успешных проектов на примере DW/BV. Корпоративные базы данных-2011. Материалы 16-й ежегодной технической конференции // 2011. – CitForum.ru

3. Писаренко Д.С., Рублев В.С. Объектная СУБД Динамическая информационная модель и ее основные концепции // Моделирование и анализ информационных систем – 2009. – Т.16, №1, С. 62-91

4. Рублев В.С. Язык объектных запросов динамической информационной модели DIM // Моделирование и анализ информационных систем. Т. 17 (3), 2010. – С. 144-161.

5. Рублев В.С., Запросная полнота языка ODQL динамической информационной модели DIM // Ярославский педагогический вестник. Серия "Физико-математические и естественные науки Вып.1. - Ярославль - 2011 - С. 69 – 75.

6. Рублев В.С., Организация выполнения объектных запросов в динамической информационной модели DIM // Моделирование и анализ информационных систем, т.18, № 2 // Ярославль: ЯрГУ, 2011. - С. 39-51.

# О КОДАХ КОНЕЧНЫХ МНОЖЕСТВ ТОЧЕК В ЕВКЛИДОВЫХ ПРОСТРАНСТВАХ

**Руденко А.Д. (Москва)**
*aleksei-rudenko@yandex.ru*

## Введение

В работах В.Н. Козлова ([1]) при построении геометрического подхода к задаче распознавания изображений вводится код специального вида.

Пусть $A \subset R^2$ — конечное множество. Перенумеруем точки из $A$ так, чтобы их номера были попарно различны. Обозначим через $M_A$ множество этих номеров. Пусть $S(i_0, i_1, i_2)$ — площадь треугольника с вершинами в точках с номерами $i_0, i_1, i_2$ из $M_A$.

Введем $\rho(i_0, i_1, i_2; j_0, j_1, j_2) = \frac{S(i_0,i_1,i_2)}{S(j_0,j_1,j_2)}$ $(i_0, i_1, i_2, j_0, j_1, j_2 \in M_A$, порядок номеров в тройках не важен); если $S(j_0, j_1, j_2) = 0$, считаем, что значение $\rho(i_0, i_1, i_2; j_0, j_1, j_2)$ не определено. Обозначим через $T_A$ множество индексированных чисел $\rho(i_0, i_1, i_2; j_0, j_1, j_2)$ для всех таких пар троек. Тогда кодом $A$ назовем пару $< M_A, T_A >$.

Назовем конечные множества $A, B \subseteq R^2$ эквивалентными, если существует такая биекция

$$\psi \colon M_A \to M_B \colon \forall i_0, i_1, i_2, j_0, j_1, j_2 \in M_A \colon$$
$$\rho(i_0, i_1, i_2; j_0, j_1, j_2) = \rho(\psi(i_0), \psi(i_1), \psi(i_2); \psi(j_0), \psi(j_1), \psi(j_2));$$

Оказываются верны следующие утверждения:

**Теорема 1.** *Если конечные множества $A, B \subset R^2$ аффинно эквивалентны, то они эквивалентны.*

Будем называть конечное множество точек плоскости плоским изображением, если оно не лежит целиком ни на какой паре параллельных прямых.

**Теорема 2.** *Плоские изображения аффинно эквивалентны тогда и только тогда, когда эквивалентны.*

Аналогичным образом вводится код конечного множества из $R^3$ (вместо троек точек рассматриваются четверки, вместо треугольников — тетраэдеры, вместо прямых — плоскости), и аналоги теорем 1 и 2 оказываются верны.

Логично предположить, что похожие построения можно провести и в пространстве $R^n$ произвольной размерности с введенными

стандартным образом скалярным произведением и мерой. Здесь рассматривается эта задача.

## Понятие кода множества

Через $V(a_0, \ldots, a_n)$ бозначим меру $n$-мерного симплекса с вершинами в точках $a_0, \ldots, a_n$ из $R^n$. Пусть $\Phi\colon (R^n)^{2n+2} \to R \cup \{\Delta\}$,

$$\Phi(a_0, \ldots, a_n; b_0, \ldots, b_n) = \begin{cases} \frac{V(a_0,\ldots,a_n)}{V(b_0,\ldots,b_n)}, V(b_0, \ldots, b_n) \neq 0, \\ \Delta, V(b_0, \ldots, b_n) = 0; \end{cases}$$

**Определение.** *Пусть $A, B \subset R^n$ конечны, $g\colon A \to B$ — биекция. Будем говорить, что $A$ эквивалентно $B$ с отображением $g$, если для любых точек $a'^0, \ldots, a'^n, a''^0, \ldots, a''^n$ из $A$ выполняется*

$$\Phi(a'^0, \ldots, a'^n; a''^0, \ldots, a''^n) = \Phi(g(a'^0), \ldots, g(a'^n); g(a''^0), \ldots, g(a''^n));$$

*Будем говорить, что коды множеств $A$ и $B$ равны, если существует такая биекция $g$, что $A$ эквивалентно $B$ с отображением $g$.*

**Определение.** *Конечное множество в $R^n$, не лежащее целиком ни в какой паре параллельных гиперплоскостей, называется $n$-мерным изображением.*

Для $n$-мерных изображений оказывается верна теорема 7 — аналог теоремы 2. Далее строится схема её доказательства.

## Вспомогательные понятия и утверждения

**Определение.** *Пусть $A \subset R^n$ — конечное множество, $g\colon R^n \to R^n$ — биекция. Назовем пару $(A, g)$ неправильной, если*

1. *$e^i \in A, g(e^i) = e^i, i = \overline{0, n}$, здесь $e^1, \ldots, e^n$ — векторы стандартного базиса, $e^0 = 0$,*
2. *$\exists a \in A\colon g(a) \neq a, \sum_{i=1}^n a_i \neq 1$,*
3. *множества $A$ и $g(A)$ эквивалентны с отображением $g$.*

Далее везде в этом параграфе $(A, g)$ — неправильная пара.

**Лемма.** *Для каждой точки $a$ множества $A$, $|a_i| = |g(a)_i|, i = \overline{1, n}$.*

Таким образом, положив

$$E_g(a;-1) = \{i \in [1,n] | a_i = -g(a)_i \neq 0\},$$
$$E_g(a;0) = \{i \in [1,n] | a_i = g(a)_i = 0\},$$
$$E_g(a;1) = \{i \in [1,n] | a_i = g(a)_i \neq 0\},$$

получим

$$[1,n] = E_g(a;-1) \oplus E_g(a;0) \oplus E_g(a;1);$$

**Лемма.***Пусть $a$ — точка множества $A$, $g(a) \neq a$. Тогда либо при $\alpha = -1$, либо при $\alpha = 1$, множество $E_g(a;\alpha)$ не пусто, $\sum_{i \in E_g(a;\alpha)} a_i = \frac{\alpha+1}{2}$.*

**Определение.***Пусть $l \geq 0$ — целое число, $(A,g)$ — неправильная пара, $a^0, \ldots, a^l$ — точки множества $A$, $\alpha^i \in \{-1,1\}, i = \overline{0,l}$. Будем говорить, что $[(a^0;\alpha^0), \ldots, (a^l;\alpha^l)]$ — путь по $(A,g)$ длины $l$ из $(a^0;\alpha^0)$ в $(a^l;\alpha^l)$ если либо $l = 0$, либо $l > 0$ и для каждого целого $i$ из $[1,l]$ пересечение множеств $E_g(a^{i-1};\alpha^{i-1})$ и $E_g(a^i;\alpha^i)$ не пусто.*

**Теорема 3.***Пусть $(A,g)$ — неправильная пара. Ни для какой точки $a$ множества $A$ не существует пути по $(A,g)$ из $(a;\alpha)$ в $(a;-\alpha)$, $\alpha \in \{-1,1\}$.*

**Теорема 4.***Пусть $(A,g)$ — неправильная пара, $b,c$ суть точки множества $A$, причем $b \neq g(b), \sum_{i=1}^n b_i \neq 1$. Пусть также*

$$\beta, \gamma \in \{-1,1\}, E_g(b;\beta), E_g(c;\gamma) \neq \emptyset,$$
$$\sum_{i \in E_g(b;\beta)} b_i = \frac{\beta+1}{2}, \sum_{i \in E_g(c;\gamma)} c_i \neq \frac{\gamma+1}{2};$$

*Тогда не существует пути по $(A,g)$ из $(b;\beta)$ в $(c;\gamma)$*

## Свойства отношения равенства кодов

**Теорема 5.***Отношение равенства кодов конечных множеств из $R^n$ является отношением эквивалентности.*

**Теорема 6.***Если конечное множество $B \subset R^n$ переводится в конечное множество $C \subset R^n$ аффинным преобразованием $\phi$, $B$ эквивалентно $C$ с отображением $\phi$.*

**Теорема 7.***$n$-мерные изображения аффинно эквивалентны тогда и только тогда, когда их коды равны.*

Приведем короткую схему доказательства этой основной теоремы. Пусть коды двух не аффинно эквивалентных конечных множеств $B, C \subset R^n$ равны. Тогда найдётся такая неправильная пара

$(A, g)$, что $B$ аффинно эквивалентно $A$, $C$ аффинно эквивалентно $g(A)$. Возьмем такие $a \in A, \alpha \in \{-1, 1\}$, что

$$a \neq g(a), \sum_{i=1}^{n} a_i \neq 1, E_g(a; \alpha) \neq \emptyset, \sum_{i \in E_g(a;\alpha)} a_i = \frac{\alpha + 1}{2};$$

Пусть $E$ — объединение $E_g(b; \beta)$ по всем $(b; \beta)$, в которые существует путь из $(a; \alpha)$. Тогда из теорем 3, 4 следует, что $A$ лежит в паре гиперплоскостей $\sum_{i \in E} x_i = 0$, $\sum_{i \in E} x_i = 1$.

Выражаю благодарность В.Н. Козлову за научное руководство.

**Литература**

1. Козлов В. Н. Введение в математическую теорию зрительного восприятия. – М.: Изд-во Центра прикл. иссл. при мех.-мат. факультете МГУ, 2007.

2. Кудрявцев В. Б., Гасанов Э. Э., Подколзин А. С. Введение в теорию интеллектуальных систем. – М.: Изд-во ф-та ВМиК МГУ, 2006.

3. Ильин В. А., Ким Г. Д. Линейная алгебра и аналитическая геометрия. – М.: Изд-во Проспект, 2008.

# СИСТЕМЫ ОЦЕНКИ И МОНИТОРИНГА СЛОЖНЫХ ПРОЦЕССОВ И ИХ ПРИЛОЖЕНИЯ

**Рыжов А.П. (Москва)**
*ryjov@mail.ru*

В работе описывается технология информационного мониторинга сложных процессов, разрабатываемая автором с конца 80-х - начала 90-х годов. Приводится содержательная постановка проблемы информационного мониторинга, описываются технологические и математические аспекты разработки систем информационного мониторинга.

Многие процессы в бизнесе, экономике, политике и других областях, называемых слабо (или плохо) формализуемыми, не возможно представить в виде набора уравнений, автоматов и других математических средств представления и анализа динамических систем, однако специалисты как-то решают задачи оценки состояния процесса и управления им. В общем виде задача заключается в оценке текущего состояния процесса на основе всей доступной информации, построении прогнозов его развития и выработке рекомендаций по управлению исходя из целей, стоящих перед специалистом. В работе приводятся примеры таких задач из политологии, маркетинга, финансового анализа, страхового дела. В качестве примера процессов, не являющихся таковыми, можно привести взаимодействие двух тел или распространение колебаний в однородной среде. Имеются математические модели таких процессов, информация измерима и доступна, результат можно вычислить для любого момента времени.

Будем называть задачу оценки текущего состояния системы (процесса) и построении прогнозов ее развития задачей информационного мониторинга, а человеко-компьютерные системы, обеспечивающие аналитическую поддержку подобного рода информационных задач, системами информационного мониторинга. Основными элементами систем информационного мониторинга являются информационное пространство и аналитик. В работе анализируются свойства информационного пространства: разнородность, фрагментарность, разноуровневость и ненадежность доступной информации, ее противоречивость и изменяемость во времени.

Приводятся архитектурные и технологические особенности компьютерных систем, обеспечивающие обработку такого рода инфор-

мации [3]. В частности:

- для реализации возможности обработки информации из разнородных источников, в базе данных системы хранятся как сами документы, так и ссылки на них с оценкой содержащейся в них информации, данной экспертом;
- для возможности обработки фрагментарной информации используется модель процесса в виде графа;
- обработка разноуровневой информации достигается за счет предоставления пользователю возможности отнести оценку конкретного информационного материала к разным вершинам модели;
- обработка информации различной степени надежности и обладающей возможной противоречивостью или тенденциозностью достигается за счет использования лингвистических оценок экспертами данной информации;
- изменяемость во времени учитывается фиксацией даты поступления информации при оценке конкретного материала, т.е. время является одним из элементов описания объектов системы.

Для эффективного практического применения предложенных технологических решений необходима проработка ряда теоретических проблем, результаты которой приводятся в докладе. Рассматриваются три такие проблемы.

**Проблема 1.** Можно ли, учитывая некоторые особенности восприятия человеком объектов реального мира и их описания, сформулировать правило выбора оптимального множества значений признаков, по которым описываются эти объекты? Возможны два критерия оптимальности:

*Критерий 1.* Под оптимальными понимаются такие множества значений, используя которые человек испытывает минимальную неопределенность при описании объектов.

*Критерий 2.* Если объект описывается некоторым количеством экспертов, то под оптимальными понимаются такие множества значений, которые обеспечивают минимальную степень рассогласования описаний.

Показано [6], что мы можем сформулировать методику выбора оптимального множества значений качественных признаков. Более того, показано, что такая методика является устойчивой, то есть возможные при построении функций принадлежности естественные маленькие ошибки не оказывают существенного влияния на выбор оптимального множества значений. Множества, оптимальные по критериям 1 и 2 совпадают.

**Проблема 2.** Можно ли определить показатели качества поиска информации в нечетких (лингвистических) базах данных и сформулировать правило выбора такого множества лингвистических значений, использование которого обеспечивало бы максимльные показатели качества поиска информации?

Показано [4], что можно ввести показатели качества поиска информации в нечетких (лингвистических) базах данных и формализовать их. Показано, что возможно сформулировать методику выбора оптимального множества значений качественных признаков, которое обеспечивает максимальные показатели качества поиска информации. Более того, показано, что такая методика является устойчивой, то есть возможные при построении функций принадлежности естественные маленькие ошибки не оказывают существенного влияния на выбор оптимального множества значений.

**Проблема 3.** Можно ли предложить процедуры выбора операторов агрегирования информации в нечетких иерархических динамических системах, минимизирующих противоречивость модели проблемы/процесса в системах информационного мониторинга?

Можно выделить следующие подходы к решению этой проблемы, базирующиеся на различных интерпретациях операторов агрегирования информации [5]: геометрический, логический и подход на основе обучения, включающий вы себя обучение на основе генетических алгоритмов и обучение на основе нейронных сетей.

В докладе описываются разработанные на базе описанной технологии системы оценки и мониторинга процессов нераспространения ядерных технологий и материалов [7], системы оценки и мониторинга рисков атеросклеротических заболеваний [1], системы оценки и мониторинга проектов в микроэлектронике [2].

**Литература**

1. Ахмеджанов Н.М., Жукоцкий А.В., Кудрявцев В.Б., Ога-

нов Р.Г., Расторгуев В.В., Рыжов А.П., Строгалов А.С. Информационный мониторинг в задаче прогнозирования риска развития сердечно-сосудистых заболеваний. Интеллектуальные системы, Т.7, вып. 1-4, 2003, с. 5 - 38.

2. Лебедев А.А., Рыжов А.П. Оценка и мониторинг проектов разработки высокотехнологических изделий микроэлектроники. Известия ТРТУ, Тематический выпуск , ISBN 5-8327-0249-2, 2006, № 8, с. 93-99.

3. Рыжов А.П. Информационный мониторинг сложных процессов: технологические и математические основы. Интеллектуальные системы, Том 11, вып. 1-4, 2008, с. 101-136.

4. Рыжов А.П. Модели поиска информации в нечеткой среде. Издательство Центра прикладных исследований при механико-математическом факультете МГУ, М., 2004, 96с.

5. Рыжов А.П. Об агрегировании информации в нечетких иерархических системах. Интеллектуальные системы, Том 6, Вып. 1-4, 2001, с. 341.

6. Рыжов А.П. Элементы теории нечетких множеств и измерения нечеткости. М., Диалог-МГУ, 1998, 116 с.

7. A. Fattah, V. Pouchkarev, A.Belenki, A.Ryjov and L.A. Zadeh. Application of Granularity Computing to Confirm Compliance with Non-Proliferation Treaty. In: Data Mining, Rough Sets and Granular Computing. Ed. by Tsau Young Lin, Yiyu Y. Yao, L.A. Zadeh. Physica-Verlag Heidelberg, 2002, p. 308-338.

# ЭФФЕКТИВЫЕ РЕАЛИЗАЦИИ СТРОКОВЫХ СЛОВАРЕЙ ДЛЯ РЕШЕНИЯ ЗАДАЧ КОМПЬЮТЕРНОЙ ЛИНГВИСТИКИ

**Скатов Д.С. (Нижегородский государственный университет им. Н.И.Лобачевского)**
*ds@dictum.ru*

**Введение**

В настоящем исследовании нас будут интересовать реализации словарей для хранения строк и ассоциированных с ними значений. Рассматривается 5 библиотек на C/C++ из открытых репозиториев, две авторских реализации на основе trie-дерева и хэширования, а также эскизная реализация алгоритмов из [6].

Для отобранных реализаций выполнены эксперименты по индексации $3 \cdot 10^6$ запросов к ПС Яндекс. Результаты позволяют решить инженерную задачу: (1) выбрать одну из существующих реализаций, либо (2) принять целесообразность создания собственной. Приведена оценка результатов и перспективы дальнейшего развития представленных авторских наработок.

**1. Свойства словарей**

Дан алфавит $A$ и $K \subseteq A^+$, $|K| < \infty$, $\forall w \in K\ \exists val\,(w) \in \mathbb{Z}$. На символах $A$ задано отношение непосредственного следования $succ\,(a, b)$, $a, b \in A$ , индуцирующее лексикографический порядок на $A^+$: $v \prec w$, $v, w \in A^+$.

Интересующие нас функции словаря:

1. По $x \in A^+$ узнать: $x \in K$ и $val\,(x)$, если $true$ (поиск по ключу);
2. По $pr = a_1 a_2 \dots a_{|pr|}$ найти все $y = pr \cdot a_{|pr|+1} \dots a_{|y|} \in K$ , $a_j \in A$, $j = \overline{1, |y|}$ (префиксный поиск);
3. По такому же $pr$, для которого есть аналогичный $y \in K$, найти ближайший справа $pr' \succ pr$ , так что $\exists\, y' = pr' \cdot a_{|pr'|+1} \dots a_{|y'|} \in K$ (итерируемость).

В (1) ключ можно интерпретировать как цепочку байтов, что допускает любую известную реализацию (хэш-таблицы, бинарные деревья и пр.). Функция (2) реализуема на словаре, допускающем выборку диапазона. Действительно, для $pr = a_1 \dots a_n$ можно взять

правого соседа $pr' \succ pr$ (для $A = \{a, b, c\}$, $pr = abc \Rightarrow pr' = abaa$) и извлечь диапазон $[pr, pr')$. Реализация (3) достижима представлением словаря графом переходов. Он может быть затем расширен до конечного автомата (КА), размечающего неиндексированный текст вхождением ключей (напр., по схеме Ахо-Корасик).

**2. Исследуемые реализации**

**Trie-дерево.** Из 5 реализаций были отобраны [1] (A) и [2] (B) как обнаружившие меньше сбоев. К сравнению была добавлена авторская реализация (C) trie, применяемая в промышленных приложениях (напр. [5]), со следующими свойствами: хранение набора значений по каждому ключу, компактная сериализация, десериализация прямым отображением файла в память. Все реализации итерируемы.

**Judy-массив.** В 2004-м году была предложена trie-подобная реализация словаря [3] с операцией (2), оптимизированная технически: использованы десятки приёмов сжатия для уменьшения числа кэш-промахов, но результирующие алгоритмы обработки весьма сложны. Для тестирования была выбрана не [3], а более простая для повторного использования и несериализуемая [4] (D).

**DAWG-граф.** DAWG (E) является сжатой формой trie-дерева. «Прямые» алгоритмы его построения, основанные на минимизации trie как КА, трудно применимы из-за расхода памяти. Улучшение заключается в онлайн-минимизации, но требует, чтобы последовательность входных терминов была отсортирована. Современные результаты дают более сложный алгоритм, но с допуском произвольного порядка. Известная реализация [8] трудно используема повторно и даёт сбои при количестве ключей, большем $2 \cdot 10^6$. Для экспериментов была построена эскизная реализация алгоритмов из [6].

*Исправление опечаток* состоит в том, чтобы по заданному $\tilde{w} \in A^+$ извлечь из словаря правильных слов $K$ варианты его исправления: $Corr_\varepsilon(\tilde{w}) = \{w \in K : \rho(w, \tilde{w}) < \varepsilon\}$. Операциям (вставка, замена, транспозиция) назначаются веса, а расстояние $\rho(w, \tilde{w})$ вычислимо алгоритмом Левенштейна-Дамерау. Его сложность $O\left(|w|^2\right)$ можно улучшить методом ветвей и границ на словаре для $K$ с функцией (3). Большее улучшение, до $O(|w|)$, достижимо, если при фиксированном $\varepsilon$ на основании весовых матриц получить объединение шаров

$$Corr_\varepsilon^{-1}(K) = \cup_{w \in K} \{\tilde{w} | \rho(\tilde{w}, w) < \varepsilon\}$$

с дальнейшим сохранением в словарь. От него требуется только (2), но из-за огромного объёма данных хранение рационально в DAWG. Алгоритм построения DAWG не должен зависеть от порядка терминов (в силу сложности генерации шаров).

**Хэширование.** Предлагается гибридная реализация (F) хэш-словаря с возможностью (2). Он представлен матрицей из $\max\{|w|\} \times |A|$ слотов, в каждом хранятся записи для ключей и префиксов. Выбор слота определяется последней буквой термина. Используется конкатенация значений хэш-функций из [7]. Другие, известные, реализации не рассматривались в силу отсутствия (2).

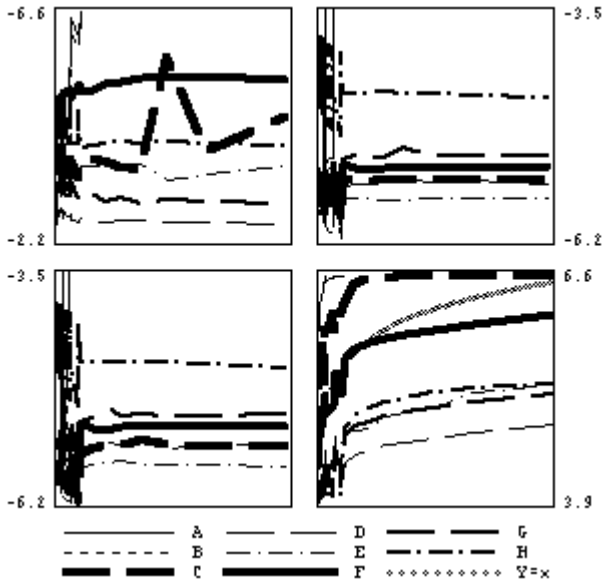



Рис. 1: Замеры (Intel Core Quad Q8200, Visual C++ 2010 x64) в шкале $\log_{10}x$. Даны оценки среднего времени для операций с одним запросом при уже имеющихся в словаре (в количестве от 1000 до 3 млн.): верхний левый (ВЛ) — добавление, ВП — успешный поиск, НЛ — неуспешный. НП — макс. объём памяти в Мб в сеансе добавления.

**SQLite DB, `std::map`.** В классе `std::map` (G) на основе красно-чёрного бинарного дерева функция (2) реализуема `lower_bound(pr)`

и `upper_bound(pr')`. `std::hash_map` не тестировалась, но, по опыту автора, в среднем она слабее trie и хэш-схем. В SQLite (H) выбор диапазона осуществляется по первичному ключу, с `synchronous=OFF` и `:memory:` (сохранение базы в память вместо файла).

**3. Эксперимент и результаты**

(A) и (B) из открытых репозиториев одинаковым образом не справились с задачей, начиная с числа запросов в словаре, превышающего $3 \cdot 10^5$.

Авторская (C) и judy (D) одинаково производительны, причём (D) не сериализуемо, а (C) требует больше памяти для построения. Хэш-схема (F), несмотря на тривиальную реализацию слотов на `std::map`, показывает лучшую в сравнении с ним производительность. Дальнейшие улучшения могут быть достигнуты применением техник из (D) в (C) и оптимизацией хранения в (F).

(H) (несмотря на `:memory:`) не является производительной реализацией, однако (G) может стать хорошим вариантом на начальных этапах — несмотря на доступность, его нельзя рассматривать в качестве худшей границы для оценок производительности. Лидером стала эскизная реализация DAWG, построенная на базе [6]. Предполагается дальнейшее улучшение этого кода с использованием приёмов из (C) и (D) и реализация алгоритма, не зависящего от порядка входных терминов.

**Литература**


1. `http://opensource.jdkoftinoff.com/jdks`.
2. `http://c-algorithms.sourceforge.net`.
3. `http://judy.sourceforge.net`.
4. `judyarray.googlecode.com`.
5. Скатов Д. С., Ливерко С. В., Вдовина Н. А., Окатьев В. В. Язык описания правил в системе лексического анализа ЕЯ текстов DictaScope Tokenizer. // Труды Международной конференции Диалог'2010 — М.: Наука, 2010.
6. Carrasco R. C., Forcada M. L. Incremental construction and maintenance of minimal finite-state automata. — Computational Linguistics, 28(2), p.207–216, june 2002.
7. `http://www.burtleburtle.net/bob/hash`.
8. `www.pg.gda.pl/~jandac/fsa.html`.

## КРИТЕРИЙ СВОДИМОСТИ ЗАДАЧИ О ПРЕДОТВРАЩЕНИИ СТОЛКНОВЕНИЙ К ЗАДАЧЕ О ПРОКАЛЫВАНИИ

**Снегова Е.А. (МГУ имени М.В. Ломоносова)**
*lenasnegova@gmail.com*

В работе исследуется задача о поиске движущихся объектов, которые могут столкнуться с движущимся объектом-запросом, где под столкновением понимается нахождение объектов в опасной близости.

Опишем задачу о предотвращении столкновений, имеющую несколько параметров:

1. параметр $\rho \in (0, \frac{1}{2})$, обозначающий *расстояние опасной близости* по Манхэттену,
2. аналитическая на всей области определения строго возрастающая функция $f : \mathbb{R}^+ \to [-\rho, 1+\rho]$, такая что $f(0) = -\rho$, называемая *законом движения объектов*.
3. аналитическая на всей области определения строго возрастающая функция $f_q : \mathbb{R}^+ \to [-\rho, 1+\rho]$, такая что $f_q(0) = -\rho$, называемая *законом движения запроса*,
4. множество объектов $V(t)$, находящихся в момент времени $t$ в области $[-\rho, 1+\rho] \times [0, 1]$.

Четверку $(f, f_q, \rho, V(t))$ назовем *задачей о предотвращении столкновений*.

Предполагаем, что объект $o_i = (t_i, y_i)$ появляется на границе прямоугольника $[-\rho, 1+\rho] \times [0, 1]$ в момент времени $t_i$ в точке с координатами $(-\rho, y_i)$ и движется по закону движения $f$ параллельно оси $x$, то есть в момент $t \in [t_i, t_i + \tau_{max}]$ объект $o_i$ находится внутри прямоугольника $[-\rho, 1+\rho] \times [0, 1]$ в точке с координатами $(f(t - t_i), y_i)$.

*Запросом* $q$ назовем пару $(t_q, x)$, где $t_q \in \mathbb{R}$, а $x \in [0, 1]$.

Предполагаем, что запрос $q = (t_q, x)$ появляются на границе прямоугольника $[0, 1] \times [-\rho, 1+\rho]$ в момент времени $t_q$ в точке с координатами $(x, -\rho)$ и движется по закону движения $f_q$ параллельно оси

$y$, то есть в момент $t \in [t_q, t_q + \tau^q_{max}]$ запрос $q$ находится внутри прямоугольника $[0,1] \times [-\rho, 1+\rho]$ в точке с координатами $(x, f_q(t - t_q))$.

Скажем, что объект $o_i = (t_i, y_i)$ и запрос $q = (t_q, x)$ *сталкиваются*, если существует момент времени $t \in \mathbb{R}$, такой что

$$|f(t - t_i) - x| + |y_i - f_q(t - t_q)| \leq \rho.$$

В задаче требуется для произвольного запроса, поступившего в момент времени $t$, перечислить все объекты из библиотеки $V(t)$, с которыми он столкнется в процессе своего движения.

Основной характеристикой алгоритма решения этой задачи является сложность поиска, измеряемая в операциях вычисления значений некоторых функций, принятых за элементарные. Поскольку библиотека динамически меняется со временем, то важными характеристиками являются также сложности вставки и удаления объектов в БД. Еще одной характеристикой является объем памяти, требуемый алгоритму для хранения структур данных.

Одним из способов эффективно решить задачу о предотвращении столкновений может быть сведение данной задачи к уже известным одномерным задачам. В этой работе в качестве такой задачи рассматриваются задача о прокалывании.

Задача о прокалывании имеет эффективное статическое решение – логарифмическое время поиска и линейный объем памяти [1], а в динамическом случае [2] оптимальное решение имеет сложность поиска порядка $\sqrt{n}$, сложность вставки/удаления порядка $\log^2 n$, и использует память порядка $n$, где $n$ есть число объектов в области наблюдения.

В [3] рассматривался случай фиксированных скоростей объектов, то есть $f(t) = vt$, а $f_q(t) = v_q t$, в [4] рассматривался случай, когда $f'(t + t') - f'_q(t) \leq 0$ для любого $t \in [0, \tau^q_{max}]$ и любого $t'$, такого, что $t + t' \in [0, \tau_{max}]$. В обоих случаях задача решалась путем сведения задачи о предотвращении к задаче одномерного интервального поиска, которая имеет логарифмическую относительного общего числа объектов в библиотеке сложность поиска вставки и удаления, и линейный объем памяти. В [5], [6] приводились критерии сводимости

задачи о предотвращении столкновений к задаче одномерного интервального поиска и к задаче о прокалывании, соответственно, для случая произвольных (не обязательно аналитических) законов движения объектов и запросов. В [7] приводился критерий сводимости к одномерным задачам для случая, когда законы движения – многочлены.

*Задачей о прокалывании* назовем пару $(\mathbb{R}, Z)$, где библиотека $Z$ есть конечное множество всех интервалов с концами из $\mathbb{R}$, а $\mathbb{R}$ есть множество всех действительных чисел. Содержательно эта задача состоит в том, чтобы для произвольного запроса $p \in \mathbb{R}$ перечислить все те и только те отрезки из $Z$, которые содержат $p$.

*Ответ на запрос* $p \in \mathbb{R}$ при библиотеке $Z$ в задаче о прокалывании есть множество $J(p, Z) = \{z \in Z : p \in z\}$.

Будем говорить, что задача о предотвращении столкновений $(f, f_q, \rho, V(t))$ *сводится* с задаче о прокалывании, если существуют такие отображения $\varphi, \varphi_1, \varphi_2 : \mathbb{R} \times [0,1] \to \mathbb{R}$, что для любой библиотеки $V(t)$, любого запроса $q = (t_q, x)$ и любого объекта $o \in V$ верно

$$o \in J(f, f_q, \rho, V, q) \Leftrightarrow [\varphi_1(o), \varphi_2(o)] \in J(\varphi(q), Z),$$

где $Z = \{[\varphi_1(o), \varphi_2(o)] : o \in V\}$.

**Теорема.** *Задача о предотвращении столкновений* $(f, f_q, \rho, V(t))$ *сводится к задаче о прокалывании тогда и только тогда, когда выполнено хотя бы одно из следующих двух условий*

**Условие 1.**

1. *Если* $(x, y) \in [\rho, 1+\rho] \times [0,1] : f_q^{-1}(y) - f^{-1}(x) \le 0$, *то* $(f_q^{-1}(y))' \le (f^{-1}(x))'$;
2. *Если* $F_L(x, 0) \le 0$, *то* $\rho \in \arg\min\limits_{\xi \in [0,\rho]} F_L(x, 0, \xi)$;
3. *Если* $F_R(x, 1) \le 0$, *то* $-\rho \in \arg\max\limits_{\xi \in [-\rho, 0]} F_L(x, 1, \xi)$;
4. *Если* $(x, y) \in [0, 2\rho] \times [0,1] : F_R(x, y) \le 0$, *то* $-\rho \in \arg\max\limits_{\xi \in [-\rho, 0]} F_R(x, y, \xi)$;

5. *Если* $(x, y) \in [0, \rho] \times [0, 1] : F_L(x, y) \leq 0$, *то*
$\rho \in \arg \min\limits_{\xi \in [0,\rho]} F_L(x, y, \xi)$;

**Условие 2.** $f$ *– линейный многочлен.*

Автор благодарит своего научного руководителя профессора Гасанова Э.Э. за постановку задачи и помощь в работе.

**Литература**

1. L. Arge, M. de Berg, H. J. Haverkort, and K. Yi. The Priority R-Tree: A Practically Efficient and Worst-Case-Optimal R-Tree. Symp. of the ACM Special Interest Group on Management of Data (SIGMOD), Paris, 2004, pages 347–358.

2. Arge, L.; Vitter, J.S. *Optimal dynamic interval management in external memory*, Foundations of Computer Science, 1996

3. Скиба Е.А. Логарифмическое решение задачи об опасной близости.// Интеллектуальные системы. 2007, том 11, вып. 1–4. С.693–719.

4. Снегова Е.А. Случай задачи об опасной близости, сводящийся одномерному интервальному поиску // Интеллектуальные системы. 2009. Т. 13, вып. 1-4.

5. Снегова Е.А. Критерий сводимости задачи об опасной близости одномерному интервальному поиску // Журнал "Дискретная математика в печати.

6. Снегова Е.А. Критерий сводимости задачи об опасной близости к одномерной задаче о прокалывании. // Интеллектуальные системы. 2011.

7. Снегова Е.А. Критерий сводимости задачи об опасной близости к одномерным задачам для полиномиальных законов движения // Интеллектуальные системы, 2011.

# О КОНСТРУКТИВНОЙ ХАРАКТЕРИЗАЦИИ ПОРОГОВЫХ ФУНКЦИЙ, ИНВАРИАНТНЫХ ОТНОСИТЕЛЬНО ГРУПП ПЕРЕСТАНОВОК

**Соколов А.П. (Московский государственный университет им. М.В.Ломоносова)**
*sokolov@intsys.msu.ru*

Пороговые функции алгебры логики являются математической моделью нейронов. Они представляют интерес благодаря своим универсальным вычислительным возможностям, а также благодаря возможности их обучения. В качестве средства задания пороговых функций в работе рассматриваются линейные формы вида $x_1w_1 + ... + x_nw_n - \sigma$ с целочисленными коэффициентами и свободным членом.

В работе рассматриваются классы пороговых функций, инвариантных относительно групп перестановок. Получено полное описание данных классов в терминах групп перестановок, сохраняющих разбиение на множестве переменных.

Исследуется сложность преобразования одной пороговой функции, заданной линейной формой, к другой, путем пошагового изменения коэффициентов линейной формы. В качестве меры сложности данного процесса принимается изменение коэффициента или свободного члена линейной формы на единицу. Данный процесс может интерпретироваться как процесс обучения нейрона с пороговой функцией активации.

В работе [1], для характеризации сложности обучения в классе всех пороговых функций исследовалась шенноновская функция $\rho(n)$. Она говорит о том, сколько минимально достаточно выполнить единичных модификаций исходной линейной формы от $n$ переменных для задания желаемой пороговой функции. Было показано, что при стремлении $n$ к бесконечности величина $\log_2 \rho(n)$ растет по порядку как $n \log_2 n$. В работе [2] показано, что для почти всех пар пороговых функций от $n$ переменных сложность перестройки одной пороговой функции, заданной линейной формой, в другую с ростом $n$ растет экспоненциально.

В данной работе рассмотрен вопрос о сложности взаимной перестройки внутри классов пороговых функций, инвариантных относительно групп перестановок, заданных на множестве переменных.

Получены верхняя и нижняя оценки сложности перестройки в худшем случае.

Пусть $U = \{u_1, u_2, ...\}$ - счетный алфавит переменных. Каждое из переменных $u_i$ может принимать значения из множества $E_2 = \{0, 1\}$. В дальнейшем во избежание употребления сложных индексов мы будем использовать для обозначения букв алфавита $U$ метасимволы $x_i$ с индексами или без них.

Введем определения линейной формы и пороговой функции. *Линейной формой* назовем функцию вида

$$l_{\vec{w},\sigma}(x_1, ..., x_n) = \sum_{i=1}^{n} x_i w_i - \sigma,$$

где $w_i$ и $\sigma$ суть целые числа при $i = 1, ..., n$. Вектор $\vec{w} = (w_1, ..., w_n)$ называют вектором весовых коэффициентов, а $\sigma$ - порогом. Для простоты иногда будем обозначать линейную форму $l_{\vec{w},\sigma}(x_1, ..., x_n)$ соответствующим набором весовых коэффициентов и порога -
$(w_1, ..., w_n, \sigma)$.

Функция $f(x_1, ..., x_n) : E_2^n \to E_2$, называется *пороговой*, если существует линейная форма $l_{\vec{w},\sigma}(x_1, ..., x_n) = x_1 w_1 + ... + x_n w_n - \sigma$ такая, что

$$f(x_1, ..., x_n) = \begin{cases} 1, \text{ если } \sum\limits_{i=1}^{n} x_i w_i - \sigma \geq 0; \\ 0, \text{ иначе.} \end{cases}$$

В этом случае говорим, что *линейная форма $l_{\vec{w},\sigma}$ задает пороговую функцию* $f(x_1, ..., x_n)$, и записывается это так:
$l_{\vec{w},\sigma} \to f(x_1, ..., x_n)$, или просто $f_{\vec{w},\sigma}$. Множество всех пороговых функций от $n$ переменных $x_1, ..., x_n$ обозначим $T^n$.

Рассмотрим перестановку $\pi \in S_n$, где $S_n$ - группа перестановок над множеством $\Omega_n = \{1, ..., n\}$. Будем называть пороговую функцию $f$ *инвариантной относительно перестановки* $\pi$, если $f(x_1, ..., x_n) = f\left(x_{\pi(1)}, ..., x_{\pi(n)}\right)$.

Легко видеть, что если $f$ инвариантна относительно перестановок $\pi_1$ и $\pi_2$, то она также инвариантна относительно перестановки $\pi_1 \cdot \pi_2$, получающейся в результате последовательного применения перестановок $\pi_2$ и $\pi_1$. Отсюда следует, что для каждой подгруппы $G$

группы перестановок $S_n$ существует множество пороговых функций $T_G^n$, инвариантных относительно перестановок из $G$. Возникает естественный вопрос: сколько существует различных классов $T_G^n$ и как они описываются?

Элементы $a, b \in \Omega_n$ назовем $\pi$-*эквивалентными*, если $a = \pi^r(b)$ для некоторого целого $r$. Отношение $\pi$-эквивалентности рабивает множество $\Omega_n$ на попарно непересекающиеся классы $O_1, ..., O_k$, которые принято называть $\pi$-*орбитами*.

Далее, пусть $R = \{R_1, ..., R_k\}$ - некоторое разбиение множества $\Omega_n$. Говорят, что элементы $a, b \in \Omega_n$ *эквивалентны относительно разбиения* $R$, если $a$ и $b$ принадлежат одному и тому же подмножеству $R_i$ разбиения $R$, и обозначается это так: $a \sim b \pmod R$.

Будем говорить, что перестановка $\pi$ сохраняет разбиение $R$, если для всякого $a \in \Omega_n$ выполнено $a \sim \pi(a) \pmod R$.

**Теорема 1.** *Если пороговая функция $f \in T^n$ инвариантна относительно перестановки $\pi \in S_n$, и $O_1, ..., O_k$ - $\pi$-орбиты, то $f$ инвариантна относительно всякой перестановки $\pi' \in S_n$, сохраняющей разбиение $O_1, ..., O_k$.*

Пусть $R = \{R_1, ..., R_k\}$ - разбиение множества $\Omega_n$. Пороговую функцию $f$ назовем $R$-симметрической, если она инвариантна относительно всякой перестановки, сохраняющей разбиение $R$. Из теоремы 1 следует, что каждый класс пороговых функций, инвариантых относительно некоторой группы перестановок, однозначно задается соответствующим разбиением $R$ множества переменных. Верно и обратное: всякому разбиению $R$ множества переменных соответствует класс $R$-симметрических пороговых функций, инвариантных относительно перестановок, сохраняющих разбиение $R$.

Переменные $R$-симметрической функции, принадлежащие одному классу эквивалентности $R_i$, будем называть *симметричными*. Рассмотрим множество $T_{m,k}$ пороговых функций от $n$ переменных, где $n \leq m \cdot k$, таких, что для каждой функции $f$ из $T_{m,k}$ существует разбиение $R = \{R_1, ..., R_k\}$ множества $\Omega_n = \{1, ..., n\}$ такое, что $\max(|R_1|, ..., |R_k|) = m$ и $f$ является $R$-симметрической. Параметр $m$ характеризует максимальный размер класса симметрии: если $m = 1$, то функция - несимметрическая, если же $m = n$, то симметрическая. Параметр $k$ характеризует число независимых классов симметрии.

Введем понятие близости между линейными формами и пороговыми функциями. Пусть $l_{\vec{w}',\sigma'}$ и $l_{\vec{w}'',\sigma''}$ - линейные формы от $n$ пе-

ременных. Расстоянием между линейными формами $l_{\vec{w}',\sigma'}$ и $l_{\vec{w}'',\sigma''}$ назовем следующую величину

$$\rho\left(l_{\vec{w}',\sigma'}; l_{\vec{w}'',\sigma''}\right) = |\sigma' - \sigma''| + \sum_{i=1}^{n} |w_i' - w_i''|.$$

Эту величину интерпретируем как необходимость сделать $\rho$ последовательных единичных изменений компонент одной линейной формы, чтобы получить другую.

*Близостью пороговых функций* $f'(x_1, ..., x_n)$ и $f''(x_1, ..., x_n)$ назовем величину

$$\rho\left(f'; f''\right) = \min_{\substack{l_{\vec{w}',\sigma'} \to f' \\ l_{\vec{w}'',\sigma''} \to f''}} \rho\left(l'; l''\right).$$

Определим величину $\rho\left(m, k\right)$ следующим образом

$$\rho\left(m, k\right) = \max_{f', f'' \in T_{m,k}} \rho\left(f', f''\right),$$

где максимум берется по всем парам пороговых функций $f', f''$ из класса $T_{m,k}$. Данная величина характеризует близость между наиболее удаленными пороговыми функциями, не более чем с $m$ взаимно симметричными переменными, и числом классов симметрии - $k$.

Имеет место следующая теорема, характеризующая сложность взаимной перестройки внутри классов пороговых функций, инвариантных относительно групп перестановок.

**Теорема 2.** *Если* $k \to \infty$, *то*

$$m \cdot 2^{\frac{1}{2} k \log k + o(k)} \leq \rho\left(m, k\right) \leq m^{k+2} \cdot 2^{\frac{1}{2} k \log k + o(k)}.$$

Автор благодарит профессора Валерия Борисовича Кудрявцева за постановку задачи, а также участников семинара «Кибернетика и информатика» за ценные обсуждения, возникавшие по ходу работы.

**Литература**

1. Соколов А. П., О конструктивной характеризации пороговых функций // Интеллектуальные системы, т.12. – М.: Изд-во МГУ, 2008.

2. Соколов А. П., О сложности взаимной обучаемости почти всех нейронов // Интеллектуальные системы, т.14. – М.: Изд-во МГУ, 2010.

# АНАЛИЗ ФОРМЫ ИЗОБРАЖЕНИЙ, ЗАДАННЫХ С ПОГРЕШНОСТЬЮ

**Чуличков А.И., Демин Д.С., Копит Т.А., Цыбульская Н.Д.**
**(МГУ имени М.В. Ломоносова, физический факультет)**
*achulichkov@gmail.com*

**Форма изображения** Под изображением понимается числовая функция $f(\cdot)$, заданная на поле зрения $X \subset R^2$, множество всех изображений образует полное линейное нормированное пространство $\mathcal{R}$. Значение $f(x)$ называется яркостью изображения $f(\cdot)$ в точке $x$ поля зрения.

Изображения одного и того же объекта или сцены, полученные при различных условиях, могут различаться радикально. Заключенная в них "часть" информации, не меняющаяся при изменении условий их регистрации, называется формой изображения [1]. Форма может быть построена, если задана операция сравнения по форме: определим класс $\mathcal{F}$ функций, преобразующих яркость изображения при изменении условий его регистрации: пусть $f \in \mathcal{R}$ — изображение сцены, тогда для любого $F \in \mathcal{F}$ изображение $F * f \in \mathcal{R}$: $F * f(x) = F(f(x))$, $x \in X$, есть изображение той же сцены, полученное при некоторых условиях наблюдения. Примерами класса $\mathcal{F}$ могут быть все (измеримые) функции, монотонные функции, полиномы степени не выше заданной и т.п.

**Определение.** Изображение $g \in \mathcal{R}$ не сложнее по форме изображения $f \in \mathcal{R}$, если существует функция $F \in \mathcal{F}$, такая, что $g = F * f$.

**Определение.** Множество $V_f \subset \mathcal{R}$ всех изображений, которые не сложнее по форме, чем изображение $f$, называется формой изображения $f$.

Если $V_f$ выпукло и замкнуто, то задача наилучшего приближения

$$\|\xi - \xi_f\| = \inf\{\|\xi - g\| \,|\, g \in V_f\}$$

разрешима для любого $\xi \in \mathcal{R}$, а если ее решение единственно, то оно определяет проекцию $\xi$ на $V_f$, $\xi_f = P_f\xi$, оператор проецирования $P_f$ в этом случае связан с $V_f$ взаимно однозначно и также называется формой изображения $f$.

**Задачи, решаемые методами морфологического анализа изображений** Пусть проектор $P_f : \mathcal{R} \to \mathcal{R}$ существует. В терминах $P_f$ решается ряд важных задач анализа изображений, в частности — следующие [1].

- Пусть требуется определить, сравнимо ли предъявленное изображение $\xi$ по форме с $f$. Ответ на этот вопрос положителен, если и только если $\xi = P_f\xi$.

- Пусть имеется набор форм $V_1, \ldots, V_M$, и требуется определить, к какой форме из заданных наиболее близко (по форме) предъявленное изображение $\xi$. Ответ: к форме с номером $k_*$, где $\|\xi - P_{k_*}\xi\| \le \|\xi - P_{k_*}\xi\|$, $k = 1, \ldots, M$. Если таких номеров несколько, то ответ неоднозначен.

- Пусть требуется указать, чем отличается предъявленное изображение $\xi$ от $f$ по форме. Ответ дает изображение $\xi - P_f\xi$.

Для решения этих задач априори требуется задать форму как множество $V$ (или проектор $P$).

Однако на практике часто информация о форме содержится в изображении сцены, заданном с погрешностью или в наборе изображений одной и той же сцены, полученных при различных и неконтролируемых условиях наблюдения и искаженных шумом. В работе рассматриваются морфологические методы, основанные на форме, оцененной из неточно заданных изображений сцены.

**Сравнение по форме изображений, заданных с погрешностью.** Пусть заданы изображения $\xi, \eta \in \mathcal{R}$, полученные по схеме

$$\xi = f + \nu, \quad \eta = g + \mu,$$

где неискаженные шумом $\nu$ и $\mu$ соответственно изображения $f$ и $g$ ненаблюдаемы. Требуется по заданным $\xi$ и $\eta$ определить, сравнимо ли $g$ по форме с $f$, т.е. найдется ли такая функция $F \in \mathbf{F}$, что $g = F * f$, если $\|\nu\| \le \varepsilon$, $\|\mu\| \le \varepsilon$.

Пусть $\mathcal{F}_m$ — класс монотонных преобразований $F(\cdot)$, т.е. таких, что из $x \le y$ следует $F(x) \le F(y)$, и изображения заданы в конечном числе точек своими значениями $(f_1, \ldots, f_N)$, так, что пространство $\mathcal{R}$ является $N$-мерным линейным нормированным пространством с нормой $\|f\| = \max\limits_{i=1,\ldots,N}\{|f_i|\}$. Тогда задача сравнения изображений $\xi$ и $\eta$ по форме сводится к проверке неравенства

$$\inf_{j(\cdot)} \inf_{f_1 \le f_2 \le \ldots \le f_N; g_1 \le g_2 \le \ldots \le g_N} \{Q_j(f, g)\} \le \varepsilon,$$

где

$$Q_j(f,g) = \max_{i=1,\ldots,N} \left\{ \max \left( \left| f_i - \xi_{j(i)} \right|, \left| g_i - \eta_{j(i)} \right| \right) \right\},$$

а $j(\cdot)$ — биекция множества $\{1,2,\ldots,N\}$ на себя. Метод и алгоритм решения этой задачи приведены в работах [1-3].

**Оценка формы, заданной упорядочением яркости точек поля зрения.** В евклидовом пространстве всех изображений $\mathcal{R}_N = \{f = (f_1, f_2, \ldots, f_N) : f_1, f_2, \ldots, f_N \in (-\infty, \infty)\}$ замыкание класса изображений $V_0 = \{f : f_1 < f_2 < \ldots < f_N\}$ является замкнутым выпуклым конусом и играет важную роль в приложениях.

Пусть $g = (g_1, \ldots, g_N) \in \mathcal{R}_N$ есть изображение некоторой сцены, и $j(\cdot)$ — биекция множества $\{1,2,\ldots,N\}$ на себя, такая, что $(g_{j(1)}, g_{j(2)}, \ldots, g_N) \in V_0$, однако упорядочение $j(\cdot)$ неизвестно, и его требуется установить по наблюдению последовательности изображений $\{\xi_j = F_j * f + \nu_j, j = 1,2,\ldots\}$, в которой $F_j(\cdot)$, $j = 1,2,\ldots$ — монотонно возрастающие функции, а $\nu_j \in \mathcal{R}_N$, $j = 1,2,\ldots$, независимы в совокупности, обладают независимыми в совокупности координатами, причем $\|\nu\| \le \delta$ с вероятностью 1.

При указанных условиях справедлива следующая теорема [4].

**Теорема.** *Упорядоченность координат вектора $g \in R_N$ определяется с вероятностью единица по конечному числу наблюдений $\xi_1, \ldots, \xi_n \in R_N$.*

В этой же работе [4] предложен алгоритм упорядочения координат и получены оценки вероятности ошибочного результата упорядочения. Эти результаты получены по аналогии с упорядочением значений возможностей, предложенных Ю.П.Пытьевым в работе [5].

**Форма как линейное подпространство.** Другим важным примером формы изображения является конечномерное линейное подпространство евклидова пространства $\mathcal{R}_N$ всех изображенний. Это подпространство формально может быть задано как пространство $\mathcal{R}(A)$ значений линейного оператора $A \in \mathcal{R}_n \to \mathcal{R}_N$, восстановленного эмпирически из наблюдений изображений $\xi_1, \ldots, \xi_m \in /R_N$ сцены в условиях регистрации, заданных вектором параметров $g \in \mathcal{R}_n$, принимающим в каждой тестовой ситуации с номером $j$ известное значение $g_j$, $j = 1, \ldots, m$, по схеме

$$\xi_j = Ag_j + \nu_j, \quad , j = 1, \ldots, m,$$

где $\nu_j$ — погрешность регистрации изображения в ситуации с номером $j$, $j = 1, \ldots, m$. Методы оценки пространства значений линейных операторов по данным тестовых измерений, выполненных с погрешностью, описаны в работах [6,7].

Работа выполнена при поддержке гранта РФФИ 11-07-00338.

**Литература**

1. Пытьев Ю. П., Чуличков А. И. Методы морфологического анализа изображений. — ФИЗМАТЛИТ, 2010.

2. Демин Д. С., Чуличков А. И. Сравнение формы нескольких сигналов, порожденных нелинейным монотонным преобразованием из неизвестного прообраза, и оценивание параметров их формы. // Интеллектуализация обработки информации. Сборник докладов. М.: МАКС Пресс, 2010, с. 407–409.

3. Куличков С. Н., Чуличков А. И., Демин Д. С. Морфологический анализ инфразвуковых сигналов в акустике. — М.: Изд-во "Новый Акрополь 2010.

4. Цыбульская Н. Д., Чуличков А. И. Эмпирическое упорядочение яркости пикселей изображения, задающее его форму. // Математические методы распознавания образов: 15-я Всероссийская конференция. Сб.докладов. М.:МАКС Пресс, 2011, с.444-447.

5. Пытьев Ю. П. Математические методы и алгоритмы эмпирического восстановления стохастических и нечетких моделей. // Интеллектуальные системы. Т.11. Вып. 1-4. 2007, с. 277-327.

6. Чуличков А. И., Цыбульская Н. Д., Шахбазов С. Ю. Классификация сигналов по форме, модель которой определена эмпирически. // Вестник Московского университета. Сер.3. Физика. Астрономия. №5. 2010, с. 9-13.

7. Копит Т. А., Чуличков А. И., Устинин Д. М. Интерпретация экспериментальных данных на основе кусочно-линейной аппроксимации модели измерений. // Вестник Московского университета. Сер.3. Физика. Астрономия. №5, 2010, с. 3-8.

# РАЗРАБОТКА МОДЕЛИ РЕАЛИЗАЦИИ ПРЕДМЕТНО-ОРИЕНТИРОВАННОЙ ИНТЕРАКТИВНОЙ СЕТИ

**Ширяева И.А. (Смоленский государственный университет)**
*ir.shiryaeva@gmail.com*

Анализ современного состояния организации образовательного процесса, изучение дистанционных образовательных технологий, понятия «интерактивность», а также выявление основных аспектов построения интерактивной образовательной сети позволяет сформулировать представление о модели организации обучения на базе предметно-ориентированной интерактивной сети образовательного учреждения.

Разработанная модель организации обучения на базе предметно-ориентированной интерактивной сети в качестве методологической основы имеет системный подход, рассматривающий объект изучения как целостное образование, учитывающее и приводящее в скоординированное действие все факторы и условия, существенно влияющие на него [2].

Предметно-ориентированная интерактивная сеть должна подчиняться действию следующих принципов: доступности, интерактивности, вариативности, системности и последовательности, целесообразности применения новых информационных технологий, сознательности и активности, наглядности.

Рассматриваемая модель должна отвечать требованиям нормативности, преемственности, целостности, связанности компонентов модели, адаптивности.

В процессе изучения дистанционного обучения выявлены функции предметно-ориентированной интерактивной сети, среди которых: создание единого информационного образовательного пространства ОУ, внесение инновационных изменений в образовательный процесс, повышение уровня интерактивного взаимодействия субъектов образовательного процесса, адаптивность к индивидуальным потребностям субъектов образовательного процесса и т.д.

Разработанная модель реализации предметно-ориентированной интерактивной сети (рисунок 1) включает в себя технический компонент и компонент реализации.


**Рис. 1 Модель реализации предметно-ориентированной интерактивной сети**

В логике построения предметно-ориентированной интерактивной сети технический компонент модели заключается в выборе программной оболочки, технологии дистанционного обучения, создании базы электронных образовательных ресурсов.

Компонент реализации состоит из следующих блоков: выделены нормативный, потребностно-мотивационный, содержательный, технологический и контролирующий.

Нормативный блок включает в себя правовые, теоретические и методологические основания обучения учащихся на базе предметно-ориентированной интерактивной сети, гарантирует обучение в соответствии с требованиями ГОС, а также охватывает специфические требования, предъявляемые к выпускнику образовательного учре-

ждения как современной личности, способной жить и развиваться в информационном обществе.

Потребностно-мотивационный блок модели является основным условием, побуждающим к внедрению и применению в деятельности образовательного учреждения предметно-ориентированной интерактивной сети, содержит следующие определяющие критерии: анализ существующей ситуации в образовательном учреждении, непосредственно внедрение, а также условия стимулирования деятельности пользователей.

Следующим важным критерием является непосредственно внедрение предметно-ориентированной интерактивной сети на базе определенной ранее платформы дистанционного обучения, заключающееся в непосредственно инсталляции программного продукта, его интеграции с приложениями внешних разработчиков уже используемыми в образовательном учреждении и заполнении базы данных всей необходимой информацией.

Осознание необходимости внесения изменений в образовательный процесс недостаточно для качественного использования интерактивной сети. Отсюда следует критерий стимулирования деятельности, относящийся к преподавательскому составу (учет учебной нагрузки преподавателя, организация консультаций, учет времени проведенного преподавателем в интерактивной сети и т.д.), учащимся (создание ситуаций успеха, поощрение и т.д.), родителям (создание ситуаций успеха, организация консультаций) [1].

Содержательный блок представлен теми видами образовательного контента, которые заключают в себе платформа, построенная на ее основе интерактивная сеть и дистанционный курс: интерактивные модули знаний по предмету, адаптивные тестирующие оболочки, средства интерактивной коммуникации.

Модуль знаний — основная единица дистанционного обучающего курса. При разработке модулей знаний необходимо учитывать не только то, что каждый модуль должен давать совершенно определенную самостоятельную порцию знаний по теме урока, но и сформировать необходимые умения. Отсюда следует необходимость в интерактивности, которая заключается в своевременном отслеживании достижений учащегося, взаимодействии с преподавателем или самим модулем знаний, который в случае отсутствия преподавателя в интерактивной сети должен выполнять ряд его функций.

Для соответствия требованиям объективности в оценке качества знаний учащихся обучающий курс интерактивной сети должен иметь адаптивные тестирующие оболочки, позволяющие придать большую индивидуальность дистанционному обучению.

Средства интерактивной коммуникации — обязательный элемент интерактивного дистанционного курса. Для обеспечения интерактивности в курсах дистанционного обучения целесообразно использовать форумы, чаты, службы коротких сообщений, аудио- и видеоконференции, электронную почту.

Технологический блок модели призван обеспечить процесс обучения методами, приемами, средствами и формами, способствующими наиболее полной организации дистанционного обучения, и включает в себя два блока: способы организации обучения на базе предметно-ориентированной интерактивной сети и определение средств интерактивной коммуникации.

Контролирующий блок модели направлен на мониторинг качества обучения, организованного на базе предметно-ориентированной интерактивной сети с использованием как встроенных в оболочку протоколов формирования отчетов, так и эмпирических методов исследования. Качественный мониторинг необходим на каждом этапе реализации предметно-ориентированной интерактивной сети, поэтому контролирующий блок относится к каждому рассмотренному блоку модели.

Таким образом, разработанная нами и рассмотренная модель реализации предметно-ориентированной интерактивной сети позволяет подойти к внедрению и использованию дистанционного обучения как целостному процессу, в ходе которого осуществляется целенаправленное интерактивное взаимодействие педагога, учащегося и дистанционного курса.

**Литература**

1. Галченкова И. С. Алгоритм стратегии внедрения дистанционного обучения / И.С. Галченкова // Известия Смоленского государственного университета. — 2009. - № 2(6). — С. 271 — 280.

2. Кушнер Ю. З. Методология и методы педагогического исследования (учебно-методическое пособие). — Могилев: МГУ им. А.А. Кулешова, 2001. – — 66 с.

# Секция “Теория автоматов”

# О СУПЕРПОЗИЦИЯХ АВТОМАТОВ

**Бабин Д.Н. (МГУ им. М. В. Ломоносова)**
*d.n.babin@mail.ru*

Понятие автомата относится к числу важнейших в математике. Содержательно автомат представляет собой устройство с входными и выходными каналами. На его входы последовательно поступает информация, которая перерабатывается им с учетом строения этой последовательности и выдается через выходные каналы. Эти устройства могут допускать соединение их каналов между собой. Отображение входных последовательностей в выходные называют автоматной функцией, а возможность получения новых таких отображений за счет соединения автоматов приводит к алгебре автоматных функций.

Первый толчок к возникновению теории автоматов дала работа Поста Э. 1921 года [1]. В ней были получены фундаментальные результаты о строении решетки замкнутых классов булевых функций, которые были в дальнейшем методически переработаны и упрощены в книге Яблонского С.В., Кудрявцева В.Б., Гаврилова Г.П. "Функции алгебры логики и классы Поста"[2].

Сами автоматы и их алгебры начали исследоваться в тридцатые годы текущего столетия, но особенно активно в период с 50-х годов. Основополагающую роль здесь сыграли работы Тьюринга, авторов знаменитого сборника "Автоматы"[3] Шеннона, Мура, Клини и других. Последующие работы по изучению алгебр автоматов велись под большим влиянием известных статей А.В. Кузнецова [4,5] и С.В. Яблонского [6] по теории функций $k$-значной логики.

Эти функции могут рассматриваться как автоматы без памяти, к которым применяются операции суперпозиции. Возникшие для таких функций постановки задач о выразимости, полноте, базисах, решетке замкнутых классов и другие, а также развитый аппарат сохранения предикатов как ключевой для решения этих задач, оказались весьма действенными и для алгебр автоматных функций. При этом под выразимостью понимается возможность получения функций одного множества через функции другого с помощью заданных операций, а под полнотой — выразимость всех функций через заданные.

Основу результатов для функций $k$-значной логики составля-

ет подход А.В. Кузнецова, опирающийся на понятие предполного класса. Для конечно-порожденных систем таких функций семейство предполных классов образует критериальную систему; другими словами, произвольное множество является полным точно тогда, когда не является подмножеством ни одного предполного класса. Множество этих предполных классов оказалось конечным и из их характеризации вытекает алгоритмическая разрешимость задачи о полноте.

На этом пути С.В. Яблонским путем явного описания всех предполных классов была решена задача о полноте для функций трехзначной логики, а вместе с А.В. Кузнецовым найдены отдельные семейства предполных классов для логики произвольной конечной значности. Затем усилиями многих исследователей [7–11] последовательно были открыты новые такие семейства, а заключительные построения провел Розенберг [12].

Одновременно с изучением функций без памяти ( без учета времени ), были сделаны попытки применения аппарата предполных классов в задаче полноты для автоматов. Сначала для автоматов без обратных связей, называемых функциями с задержками, В.Б. Кудрявцев эффективно решил задачу о полноте и ее естественных модификациях [13]. После этого им было проведено рассмотрение общего случая и на этом пути был получен фундаментальный результат негативного характера, который показал континуальность множества предполных классов автоматных функций [14]. В дальнейшем, Кратко М.И. была показана алгоритмическая неразрешимость задачи выразимости автоматных функций [15].

Особенностью операций с автоматами является то, что число состояний растет с ростом схемы, состояния дублируют друг друга или не достижимы вовсе из начального состояния. Но даже если учитывать недостижимые состояния проблема полноты [16] также остается алгоритмически неразрешимой.

Известно [17], что всякая полная относительно суперпозиции система автоматов бесконечна. С.В. Алешин [18] установил в каких случаях, в зависимости от мощностей алфавитов входного, выходного и состояний, существуют базисы для автоматов.

Автор [19] показал, что существуют полные системы ( естественно, бесконечные ) арности два (аналог 13 проблемы Гильберта для о.д.-функций). Более того автору удалось показать, что система, состоящая из одноместных конечных автоматов и всех булевых функ-

ций, полна относительно суперпозиции.

Как уже отмечалось, задача выразимости для конечных автоматов в общем случае алгоритмически неразрешима. Представляет интерес накопить примеры, когда эта задача имеет положительное решение.

Рассматривая автоматную функцию, трудно учесть разницу между о.д.-функцией и д.-функцией, поэтому решение задачи выразимости для конечных автоматов относительно суперпозиции путем анализа сохраняемых автоматом предикатов неизбежно наталкивается на большие сложности. Представляет интерес рассматривать системы автоматов, не сохраняющие никаких конечных предикатов, например, когда в выражающей системе есть автоматы "штрих Шеффера и задержка". В этом случае задача выразимости решается в терминах внутренней структуры рассматриваемых автоматов, а не в терминах сохраняемых предикатов.

В частности на этом пути удалось решить задачу выразимости константных автоматных функций. Летуновский А.А. [20] показал, что если конечные системы автоматов $S$ содержат "штрих Шеффера и задержку то существует алгоритм, проверяющий выразимость константной автоматной функции через систему $S$.

Определим простой автомат по аналогии со свойством простых чисел. Автомат $C$ назовем простым, если из того, что $C$ выразим через автоматы $A$ и $B$ следует, что С выразим через автомат $A$, или $C$ выразим через автомат $B$. Если рассматривать функциональную систему с операцией суперпозиции (а также композиции), то простых автоматов в таком понимании вообще нет. Операцию суперпозиции при наличии конечной добавки назовем расширенной суперпозицией, в этом случае техника разложения автоматов на простые автоматы позволяет решать некоторые задачи выразимости автоматов. В частности в этой функциональной системе можно описать все простые автоматы.

Ослабленной задачей полноты относительно суперпозиции назовем задачу выразимости всех автоматов с фиксированным числом состояний $N$. Оказывается, что для всех натуральных $N$ все автоматы с не более чем $N$ состояниями выразимы через один автомат с $N$ состояниями и двумя входами [23].

**Литература**

1.Post E., Two-valued iterative systems of math. logik.Printston, 1941.

2.Кудрявцев В.Б., Гаврилов Г.П., Яблонский С.В., Функции алгебры логики и классы Поста, Наука, М., 1966.

3. Автоматы, Сборник статей под редакцией Маккарти и Шеннона, ИЛ, Москва, 1956.

4. Кузнецов А.В., О проблемах тождества и функциональной полноты для алгебраических систем, Труды третьего всесоюзного математического съезда, т.2, М. Изд. АН СССР, 1956, с.145-146.

5. Кузнецов А.В., Структуры с замыканием и критерии функциональной полноты, Успехи математических наук т.16, N 2, 1961, с.201-202.

6. Яблонский С.В., Функциональные построения в k-значной логике, Труды Матем. ин-та им. В.А. Стеклова, АН СССР, 1958, т.51, с.5–142.

7. Ло Чжу-Кай, Предполные классы, определяемые k-арными отношениями в k-значной логике. Acta Sci. Natur. Univ. Jilinensis, 1964, N3.

8. Ло Чжу-Кай, Лю Сюй Хуа, Предполные классы, определяемые бинарными отношениями в многозначной логике,. Acta Sci. Natur. Univ. Jilinensis, 1963, N4.

9. Захарова Е.Ю., Критерий полноты системы функций из $P_k$. Проблемы кибернетики, 1967, N18, с.5–10.

10. Мартынюк В.В., Исследование некоторых классов функций в многозначных логиках.Проблемы кибернетики, 1960 N3, с.49–60.

11. Пан Юн-Цзе, Один разрешающий метод для отыскания всех предполных классов в многозначной логике, Acta Sci. Natur. Univ. Jilinensis, 1963 N3.

12. Rosenberg J., La structure des fonctions de plusiers variables sur un ensemble fini. Comptes Rendus Acad. Sci. Paris, 1965 N 260, с.3817–3819.

13. Кудрявцев В.Б., Теорема полноты для одного класса автоматов без обратных связей, Проблемы кибернетики, 1962 N 8, с.91–115.

14.Кудрявцев В.Б., О мощностях множеств предполных классов некоторых функциональных систем, связанных с автоматами, ДАН СССР т.151,N3,1963, с.493-496.

15. Кратко М.И., Алгоритмическая неразрешимость проблемы распознавания полноты для конечных автоматов, ДАН СССР, 1964, т.155, №1, с.35–37.

16. Хазбун И.В., Об условиях полноты и выразимости в точной алгебре автоматов, Логико-алгебраические конструкции, Тверь 1984, с 35-41.

17. H. H. Loomis, Jr., Completeness of sets of delayed-logic devices, IEEE Trans. Electron. Comput. EC-14 (1965), 157?172.

18.Кудрявцев В.Б., Алешин С.В., Подколзин А.С., Введение в теорию автоматов, Наука, М., 1985.

19. Бабин Д.Н., О полноте двухместных о.д.-функций относительно суперпозиции,Дискретная математика, том 1, 1989, выпуск 4, с.86-91, Наука, Москва.

20. Летуновский А.А. О выразимости константных автоматов, Интеллектуальные системы , том 9, вып. 1-4, 2005 год стр.457-469

21.М.Арбиб Алгебраическая теория автоматов языков и полугрупп, "Статистика М.,1975.

22. Алешин С.В., Об одном следствии теоремы Крона-Роудза, Дискретная математика, том 11, вып.4, 1999 год, стр. 101-109

23. Бабин Д.Н. О суперпозициях о.д.-функций ограниченного веса, Логико-алгебраические конструкции, Тверь 1984, с 21-27.

## ПРОСТЫЕ АВТОМАТЫ В ЗАДАЧЕ ПОЛНОТЫ ОТНОСИТЕЛЬНО СУПЕРПОЗИЦИИ

**Бабин Д.Н. (Москва, МГУ, механико-математический факультет)**
*d.n.babin@mail.ru*

Автоматная функция является отображением, ее задающий минимальный автомат имеет полугруппу переходов. Суперпозиция автоматов, индуцирует операцию расширения над полугруппами автоматов. Этот подход рассмотрен во многих работах [1,2,3]. Определим простой автомат по аналогии со свойством простых чисел. Автомат $C$ — простой, если из того, что $C$ выразим через автоматы $A$ и $B$ следует, что $C$ выразим через автомат $A$, или $C$ выразим через автомат $B$. Понятие простого автомата становится интересным, если при суперпозиции разрешается использовать служебные автоматы из класса $K$ — "штрих Шеффера", "задержку", а также все автоматы с безусловными переходами. В этом случае верна лемма о копировании [4], которая позволяет получать из автоматов с некоторой полугруппой автоматы с большими входными алфавитами и той же полугруппой. Простыми автоматами в этом случае будут автоматы, полугруппы которых — суть простые группы. В отличие от простых чисел, простые автоматы могут быть друг через друга выразимы. Простые знакопеременные группы $A_n$ как раз таковы, что все автоматы с не более чем $n$ состояниями выразимы через автомат с группой $A_n$. Класс $K$ не расширяется до предполного. В самом деле: если в некотором классе $R$, до которого расширяется класс $K$, есть все автоматы с группами $A_n$, то в $R$ есть все автоматы. Если нет автоматов с группой $A_n$ для $n > N$, то ввиду их простоты добавкой одного автомата их не получить. Если вместо класса $K$ использовать $K_1$ — "штрих Шеффера" и "задержку", то список простых автоматов расширяется. Простыми будут не только автоматы с простыми циклическими группами и безусловные автоматы с простыми циклическими группами [5].

### Литература

1. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов. – М.: Наука, 1985.

2. Арбиб М. Алгебраическая теория автоматов языков и полугрупп. // "Статистика" – М., 1975.

3. Алешин С.В., Об одном следствии теоремы Крона-Роудза. // Дискретная математика, том 11, вып.4, 1999, с. 101-109.

4. Бабин Д.Н. О полноте двухместных автоматных функций относительно суперпозиции. // Дискретная матетматика, том 1, выпуск 4, 1989, стр. 423-431.

5. Летуновский А.А. О выразимости константных автоматов. // Интеллектуальные системы , том 9, вып. 1-4, 2005, с. 457-469.

# АВТОМАТНАЯ МОДЕЛЬ ВЗАИМОДЕЙСТВИЯ ОРГАНИЗАЦИОННЫХ СИСТЕМ

**Богомолов С.А. (г.Саратов)**
*alexbogomolov@yandex.ru*

В сообщении рассматривается автоматная модель выбора варианта поведения в процессе управления организационной системы при конкурентном взаимодействии в зависимости от сложившихся обстоятельств. Организационные автономные управляемые системы рассматриваются в качестве динамических моделей. Рынок оборонной промышленности и продукции стратегического назначения представляет собой систему различного рода заказов.

В настоящее время процессы взаимодействия в основном приводят к проблемным (конкурентным) взаимодействиям, вызванными как сокращениями госзаказа, так и определенной самостоятельности при выборе направления развития. В этой связи возникает проблема выбора рациональной стратегии поведения, которая позволит предприятиям сохранить свой потенциал и конкурентоспособность, адаптироваться к динамике внешней среды и рынка, функционировать и развиваться согласно своему назначению и миссии.

Приведем процедуру исследования ситуаций и вывода рекомендаций по поведению организационной системы в ситуациях, когда требуется принять решение по вопросам конкурентного взаимодействия. Основой взаимодействия являются различные виды (активных или пассивных) областей, воздействие которых проявляется через воздействие как на элементы так и на структуру систем.

Динамические организационные системы обладают следующими свойствами: $R_j(j = 1, 2, ...)$, - возможность развиваться (повышать свой потенциал и конкурентоспособность); $I_j(j = 1, 2, ...)$ - возможность осуществлять информационное взаимодействие с внешней средой, в частности адекватно реагировать на негативные воздействия конкурентов; $F_j(j = 1, 2, ...)$ - возможность оказывать воздействие на внешнюю среду. Под областью понимается область взаимодействий систем, а именно совокупность ситуаций, в которых осуществляется хотя бы один из указанных типов взаимодействий. Пространство определяется как множество ситуаций, возникающих в процессе рыночных отношений между организационными системами. Элементы пространства представляют ситуации, связывающие рынок за-

казов, возможных исполнителей с их потенциалами, взаимодействиями между собой и отношениями конкуренции, партнерства, нейтралитета и другими. Таким образом, пространство - это множество ситуаций, образованных следующими элементами: организационные системы, рынок заказов и инновационных предложений, различные виды взаимодействий посредством областей. Не ограничивая общности, полагаем, что на рынке заказов имеется только две организационные системы W, V. Предполагается, что системы обладают следующими возможностями: наблюдать поведение каждой из систем, относительно интересующих заказов и инноваций, осуществлять поиск информации относительно потенциала другой, развиваться, не учитывая динамику развития другой, оказывать прямое воздействие на другую систему путем участия в конкурсе или вынося на рынок свои инновации, тем самым повышая свою конкурентоспособность. Обозначим $C_V$ - цель, выбираемая системой V. Задача управления состоит в выборе в каждой ситуации пространства s векторов $r_V(s)$, $i_V(s)$, $f_V(s)$, принадлежащих соответствующим полям, так, чтобы достичь заданной цели и удовлетворить заданному критерию. Целью может быть: достижение определенной ситуации пространства, осуществление информационного взаимодействия с внешней средой (мониторинг или маркетинговые исследования), конкурентное взаимодействие с другой системой, участвуя в конкурсе заказов. При этом необходимо удовлетворить одному из критериев $K_V$: минимизация затрат, сохранение экономического потенциала, репутации и положения. Соответственно, цель системы W обозначается как $C_W$, а критерий $K_W$. Эти цели стоят перед системами, пока не существует взаимодействие между ними (каждая из систем не имела информации о присутствии другой). Как только средствами информационного взаимодействия одна из систем обнаруживает присутствие на рынке заказов другую, которая может помешать в достижении поставленной цели, то система может менять целевую установку и в соответствии с вновь возникшей ситуацией переходит в новый режим функционирования, призванный обеспечить достижение новой цели. Другая система, в свою очередь, получив информацию о наличие конкурента поступает аналогичным образом. Переход систем в новые состояния приводит к изменениям ситуации системы в целом. В наиболее общем виде, в понятии “ситуация” содержится описание возможных действий систем, то есть предоставляется выбор одно-

го из вариантов поведения. Рассмотрим процедуру принятия решения, и уточним, как и в каких случаях происходит выбор одного из рекомендуемых вариантов поведения. В качестве моделей описания динамического поведения системы предлагается рассматривать конечные детерминированные автоматы, функционирование которых задается функциями переходов и выходов, определяющих для каждого входного сигнала и внутреннего состояния состояние в следующий тактовый момент и выходную реакцию автомата [1]. Пусть автомат $A = \{X, Y, S, \delta, \lambda\}$ [1], где $x \in X$ - входное слово (воздействие среды); $y \in Y$ - выходное слово (реакция на воздействие среды); $s \in S$ - состояние автомата. Состояние автомата $s \in S_A$ определяется следующим образом. Пусть определены дискретные множества $I = \{I_1, I_2, \ldots, I_k\}$ - система имеет информацию о внешней среде (в частности о конкуренте) в различной степени. $P = \{P_1, P_2, \ldots, P_n\}$ - варианты направления развития системы (до обнаружения конкурента, после обнаружения, противостояние с конкурентом- участие в конкурсе, уход от конкурентной борьбы - поиск других рынков, компромиссное решение, диверсификация). $F = \{F_1, F_2, \ldots, F_r\}$ - система оказывает активное воздействие с разной степенью и в различной форме с целью ослабления позиций конкурента. Содержательно, множество $I$ - различная степень неполноты информации о внешней среде, наличие конкурентов $I = \{I_1, I_2, I_3, I_4\}$, где $I_1$ - система имеет максимально полную информацию об окружающей среде (достаточной для принятия решения относительно выбора или изменения стратегии поведения во внешней среде) для выбора варианта поведения; $I_2$ - система обладает неполной информацией (недостаточной) для принятия решения относительно поведения во внешней среде; $I_3$ - система не имеет вообще информации об окружающей среде, необходимой для принятия решения относительно выбора варианта поведения, но в состоянии провести необходимые информационные воздействия с целью увеличения объема информации о конкуренте; $I_4$ - система не имеет вообще информации об окружающей среде и нет возможности ее приобрести. Пусть также определено такое множество (меры противостояния или компромиссного разрешения ситуаций, сотрудничество, переговоры) $F = \{F_1, F_2\}$: $F_1$ - оказывает воздействие на конкурирующий объект и участвует в конкурсе заказов; $F_2$ - объект не оказывает воздействие на объекты внешней среды, игнорирует предлагаемые заказы и не выходит на рынок со

своими инновациями. Определим множество $P = \{P_1P_2, P_3, P_4\}$: $P_1$ - состояние динамического покоя системы (без учета влияния внешней среды); $P_2$ - направление развития системы под действием управления, выработанного до обнаружения конкурента W во внешней среде; $P_3$ - направление развития системы (противостояние конкуренту), успеть принять меры когда конкурент уже обнаружен (возможных функциональных воздействий со стороны конкурентов); $P_4$ - направление развития, позволяющее осуществить мероприятия или действия (маневр), с целью избежать прямого конфликта в состоянии, допускающем воздействие со стороны конкурента (меры, принимаемые или направленные на минимальное взаимодействие с конкурентами). Границы областей определяется совокупностью предикатов (условий). Таким образом, каждое состояние $s_i$ описывается тройкой:

$$s_j = \{I_i, F_n, P_e\} n = 1, 2; e = 1, 2, 3, 4; I = l, 2, 3.$$

Входными сообщениями (символами) полагаем следующие сообщения о внешней среде: $x_1$ - W - имеет преимущества в конкуре перед V; $x_2$ - V - имеет преимущества в конкурсе перед W; $x_3$ - V - определяет наличие конкурента; $x_4$ - W - оказывает воздействие на V, с целью не допустить к конкурсу систему V и т. п. Затем формируется таблица переходов автомата, который в данном случае представляется автоматом - акцептором, имеющим два финальных состояния $s_C$ - система достигает намеченной цели; $s_O$ - система прекращает деятельность, не достигнув цели. Для примера рассмотрим следующую ситуацию. Ситуация. Взаимодействие в условиях, когда V обладает полной информацией о возможностях конкурента и его положении относительно системы V. Перед V встает выбор: избежать прямого столкновения при участии в конкурсе, потеряв при этом заказ или выйти на конкурс со своим инновационным предложением и попытаться выиграть конкурс. Таких ситуаций при конкурентном взаимодействии достаточно большое множество и, при соответствующей перенумерации состояний автомата, для каждой из них может быть построен автомат, представляющий всевозможные пути развития ситуации.

**Литература**

1. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов. – М.: Наука, 1985.

# МЕТОДЫ ИНТЕРПОЛЯЦИИ ЧАСТИЧНО ЗАДАННЫХ ЗАКОНОВ ФУНКЦИОНИРОВАНИЯ АВТОМАТОВ

**Епифанов А.С. (г.Саратов)**
*EpifanovAS@list.ru*

**Введение.**

В основополагающих работах , содержащих развитие теории автоматов (см., например, [1-4]) , не рассматривается задача доопределения автоматов на основе единого подхода. Существуют задачи, при решении которых используемые методы предполагают полностью заданные законы функционирования автоматов, а в исходных данных эти законы представлены частично. Фундаментальные математические результаты по доопределению частично заданных графиков представлены классическими методами интерполяции Ньютона, Лагранжа, Гаусса, Бесселя, Стирлинга и др. Неприменимость этих методов для частично заданных автоматов связана с символьной формой задания автоматов таблицами, матрицами, графами, системами логических уравнений и т.п. Задание числовыми структурами законов функционирования автоматов на основе представления автоматных отображений числовыми графиками, предложенное и разработанное В.А.Твердохлебовым (см.,например, [6]), позволяет использовать классические методы интерполяции в теории автоматов. В статье [5] изложены общие положения о возможности доопределения частично заданных геометрическими образами законов функционирования автоматов методами интерполяции.

В данной работе разработаны методы интерполяции для частично заданных законов функционирования автоматов, представленных геометрическими образами и использующие: базовые точки, вторые координаты которых получены сечениями геометрических образов прямыми линиями, параллельными оси абсцисс; базовые точки интерполяции, выделенные первыми элементами некоторых вершин геометрических образов. Получены оценки для сравнения по точности интерполяции методами Ньютона, Лагранжа и др. для частично заданных законов функционирования автоматов, последовательности вторых координат вершин геометрических образов которых определены числовыми последовательностями из массива The On-Line Encyclopedia of Integer Sequences (OEIS [8]). Получены оценки для автоматов с частично заданными геометрическими образами,

представляющими класс (4,2,2)-автоматов и его подклассы и класс линейных (8,2,2)-автоматов.

**Геометрические образы законов функционирования автоматов.**

Конечные детерминированные автоматы как математические модели сформировались для описания связей множеств сигналов и состояний с небольшим числом элементов. Это отражено в способах задания автоматов, основанных на явном указании функций переходов и выходов (таблицы, конечные графы, матрицы, логические уравнения с переменными, заданными на конечных множествах). Функционирование автоматов базируется на рекурсии, которая позволяет представлять как угодно большой, но только начальный, фрагмент процесса функционирования. В работе [6] В.А.Твердохлебовым разработан новый способ задания законов функционирования дискретных детерминированных динамических систем, основанный на числовых структурах. Предложенный подход позволяет задавать законы функционирования геометрическими фигурами, которые в свою очередь могут быть заданы аналитически.

Преобразование символьной формы автоматной модели в числовую структуру (геометрический образ законов функционирования автомата) включает линейное упорядочивание автоматного отображения $\rho_s = \bigcup\limits_{p \in X^*} \{(p, \lambda(s,p))\}$ для инициального автомата $A_s = (S, X, Y, \delta, \lambda, s)$, где S, X и Y - соответственно множества состояний, входных и выходных сигналов, а $\delta : S \times X \to S$ - функция переходов, $\lambda : S \times X \to Y$ - функция выходов и $s \in S$ - начальное состояние. Автоматное отображение $\rho_s$ взаимнооднозначно преобразуется в автоматное отображение вида $\rho'_s = \bigcup\limits_{p \in X^*} \{(p, \lambda'(s,p))\}$, где $\lambda'(s,p)$ - последний знак последовательности $\lambda(s,p)$ . Для преобразования множества пар $\rho_s$ и $\rho'_s$ в графики на множестве всех слов в алфавите X вводится линейный порядок $\omega_1$ (см.[6]). Упорядоченные множества пар $(\rho_s, \omega_1)$ и $(\rho'_s, \omega_1)$ дополняются линейными порядками $\omega_0$ на $Y^*$ и $\omega_2$ на Y. В результате получаем графики $(\rho_s, \omega_1, \omega_0)$ и $(\rho'_s, \omega_1, \omega_2)$. Построенные графики размещены в системе координат с осью абсцисс $(X^*, \omega_1)$ и осями ординат соответственно $(Y^*, \omega_0)$ и $(Y, \omega_2)$. Замена элементов множеств $X^*$ и Y в графике $\gamma_s = (\rho'_s, \omega_1, \omega_2)$ их номерами по порядкам $\omega_1$ и $\omega_2$ позволяет преобразовать символьный график

$\gamma_s$ в числовой график в системе координат с осью абсцисс $N^+$ и осью ординат $\{1, 2, ..., l\}$, где $|Y| = l$.

Из геометрического образа $\gamma_s$ автомата $A_s$ выделяется последовательность вторых координат точек геометрического образа, которая взаимнооднозначно соответствует полному геометрическому образу(при выбранном порядке $\omega_1$ на $X^*$). В результате произвольная последовательность элементов из конечного множества может рассматриваться как последовательность вторых координат точек геометрического образа автомата и, следовательно, как задание законов функционирования автомата.

**Методы интерполяции частично заданных законов функционирования автоматов.**

Выбор и применение метода интерполяции по смыслу соответствуют принятию и реализации гипотезы о том, что метод интерполяции, применяемый к числовому графику, представляющему частично заданный геометрический образ автомата, достаточно точно восстанавливает точки геометрического образа, т.е. достаточно точно доопределяет частично заданные законы функционирования автомата. Следовательно, обоснованность результатов, полученных с использованием выбранного метода интерполяции, сведена к обоснованию правильности гипотезы.

Исследованные инициальные автоматы вида $A_s = (S, X, Y, \delta, \lambda, s)$, представлены классами автоматов: классами $(n, m, l)$ - автоматов, где n = |S|, m = |X|, l = |Y| , и классами $(n, m, l)_d$ начальных отрезков геометрических образов длины d , определяющих автоматы из класса $(n, m, l)$ - автоматов. В данной работе проведен сравнительный анализ точности интерполяции методами Ньютона и Лагранжа, а также модифицированными методами Ньютона и Лагранжа. Модификация методов интерполяции состоит в том, что базовыми точками интерполяции являются точки геометрических образов автономных подавтоматов вида $A_1 = (S, \{0\}, Y, \delta, \lambda, s)$ и $A_2 = (S, \{1\}, Y, \delta, \lambda, s)$ (в случае $|X| = 2$).

Результаты анализа эффективности применения методов интерполяции Ньютона и Лагранжа по отношению к частично заданным геометрическими образами автономных подавтоматов автоматам классов (4,2,2)-автоматов и линейных (8,2,2)-автоматов при различных значениях длины начального отрезка геометрического обра-

за систематизированы в форме лемм. Показано, что при небольших длинах частично заданных геометрических образах законов функционирования автоматов из класса (4,2,2)-автоматов, следует использовать метод интерполяции Ньютона, а при длинах геометрических образов от 126 до 254 интерполяция методами Ньютона и Лагранжа выравнивается по точности.

Также в работе предлагается следующий критерий выбора базовых точек интерполяции: базовыми точками интерполяции для доопределения графика, представляющего частично заданные законы функционирования автомата, предлагается использовать точки, расположенные на прямых, параллельных оси абсцисс. Такие точки удобно определять экспериментально с помощью простых устройств, выделяющих только один заданный сигнал - 0 или 1. На основе такого выбора базовых точек и использования классических методов интерполяции Ньютона и Лагранжа проведен анализ эффективности доопределения частичных автоматов, законы функционирования которых заданы последовательностями вторых координат точек геометрических образов длины до 1 млн.знаков.Полученные результаты систематизированы и представлены в виде лемм.

**Литература**

1. Глушков В. М. Синтез цифровых автоматов. -М.: Физматгиз. -1962. -476с.

2. Кудрявцев В. Б., Алешин С. В., Подколзин А. С.Введение в теорию автоматов. -М.: Наука. -1985. -320с.

3. Брауер В. Введение в теорию конечных автоматов. Радио и связь, М., 1987.

4. Гилл А. Введение в теорию конечных автоматов. М.: Наука.-1966. -272с.

5. Твердохлебов В. А. Методы интерполяции в техническом диагностировании. / Ж-л "Проблемы управления". М. N2 2007. с. 28-34.

6. Твердохлебов В. А. Геометрические образы законов функционирования автоматов.-Саратов: Изд-во "Научная книга 2008. 183с.

7. Безопасность критических инфраструктур: математические и инженерные методы анализа и обеспечения. Под.ред. В.С.Харченко. Харьков. Изд-во Национальный аэрокосмический университет им. Н.Е.Жуковского. ("ХАИ").2011г. 641с.

8. www.oeis.org (дата обращения 20.07.2011)

# КЛАССЫ ФУНКЦИЙ, ВЫЧИСЛИМЫЕ АВТОМАТАМИ

**Иванов И.Е. (МГУ им. М. В. Ломоносова)**
*ilyxasunc@yandex.ru*

Согласно классификации формальных языков Хомского каждому типу языка сопоставлен акцептор — вычислительное устройство, распознающее данный язык. Для регулярных языков акцептором является конечный автомат, для контестно-свободных — автомат с магазинной памятью, для контестно-зависимых — линейно-ограниченная машина Тьюринга, для языков, порожденных грамматиками общего вида, — машина Тьюринга.

Аналогично, можно классифицировать и функции $f : \mathbb{N} \to \mathbb{N}$ по типу вычисляющего устройства. Хорошо известен класс вычислимых функций, то есть функций, вычисляемых машиной Тьюринга. Будем говорить, что детерменированный автомат (конечный или бесконечный) вычисляет функцию $f$, если для любого натурального $n$ при подаче слова вида $0^k1^{n+1}0^\infty$ автомат выдает последовательность вида $0^l1^{f(n)+1}0^\infty$, где $k, l \in \mathbb{N}$. В работе исследуеются классы функций, вычисляемых конечными автоматами, автоматами с магазинной памятью, и автоматами с 2 магазинами.

Получены следующие рузультаты.

**Теорема 1.** *Функция $f : \mathbb{N} \to \mathbb{N}$ — вычислимая конечным автоматом функция тогда и только тогда, когда с некоторого момента выполнено*

$$f(x) = ax + b(x),$$

*где $a \in \{0, 1\}$, а $b : \mathbb{N} \to \mathbb{N}$ — некоторая периодическая функция.*

**Теорема 2.** *Функция $f : \mathbb{N} \to \mathbb{N}$— вычислимая автоматом с магазинной памятью тогда и только тогда, когда с некоторого момента выполнено*

$$f(x) = a(x)x + b(x),$$

*где $a : \mathbb{N} \to \mathbb{Q}$, $b : \mathbb{N} \to \mathbb{Q}$ — некоторые периодические функции.*

**Теорема 3.** *Функция $f : \mathbb{N} \to \mathbb{N}$ вычислима автоматом с 2 магазинами тогда и только тогда, когда она является вычислимой.*

## Литература

1. А. Ахо, Дж. Ульман. Теория синтаксического анализа, перевода и компиляции. Том 1, изд-во Мир, 1978.

2. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов. М.:Наука, 1985.

# ОЦЕНКИ АВТОМАТНОЙ СЛОЖНОСТИ КЛАССОВ БУЛЕВЫХ ФУНКЦИЙ

**Кибкало М.А. (МГУ им. М.В. Ломоносова)**
*mkibkalo@gmail.com*

Определение сложности представления языков различными структурами - одна из традиционных задач теории автоматов. В случае представимости конечными автоматами под сложностью языка понимается число состояний в представляющем его приведенном автомате. В работе рассматривается сложность представления булевых функций конечными автоматами и устанавливаются точные значения и асимптотические оценки функции Шеннона для замкнутых классов булевых функций, входящих в решетку Поста.

КЛЮЧЕВЫЕ СЛОВА: булевы функции, конечные автоматы, сложность, классы Поста.

В произвольном конечном алфавите A определим класс конечных языков, содержащих слова равной длины: $\mathcal{L}_n(A) = \{L \subseteq A^n\}$. Каждой $f \in P_2^n$ можно взаимно однозначно сопоставить конечный язык $L(f) \in \mathcal{L}_n(E)$, где $E = \{0,1\}$ по следующему правилу: слово $\tilde{\alpha} = \alpha_1 \dots \alpha_n \in L(f) \Leftrightarrow f(\tilde{\alpha}) = f(\alpha_1, \dots, \alpha_n) = 1, \alpha_i \in E, i \in 1, \dots, n$.

Введем согласно [1] понятия инициального конечного автомата (ИКА) и представимости конечного языка в ИКА. Будем говорить, что ИКА $V_q = (E, Q, E, \varphi, \psi, q)$ представляет $f \in P_2^n$, если он представляет $L(f) \in \mathcal{L}_n(E)$.

Сложностью $S(V_q)$ ИКА $V_q$ назовем число состояний в нем. Автоматной сложностью булевой функции $f \in P_2^n$ назовем наименьшую сложность ИКА, представляющего $L(f) \in \mathcal{L}_n(E)$ :
$S(f,n) = min_{V_q \sim L(f)} S(V_q)$. Пусть $\mathcal{K} \subseteq P_2$ - класс булевых функций, $\mathcal{K}(n) = \mathcal{K} \cap P_2^n$. Сложностью $\mathcal{K}(n)$ (функцией Шеннона класса $\mathcal{K}$) назовем $S(\mathcal{K}, n) = max_{f \in \mathcal{K}(n)} S(f,n)$. Поскольку множество $\mathcal{K}(n)$ определяет совокупность языков из класса $\mathcal{L}_n(E)$, будем называть $S(\mathcal{K}, n)$ функцией Шеннона соответствующего класса конечных языков.

Будем пользоваться нотацией классов Поста, введенной в [2]. Асимптотическое поведение функции Шеннона автоматной сложности классов Поста $C_i, i = 1-4,\ D_1,\ D_3,\ F_i^\mu(n), i = 1,4,5,8, \mu > 1, \mu \in \mathbb{N},\ A_i, i = 1-4,\ F_i^\infty(n), i = 1-8,\ F_i^\mu(n), i = 2,3,6,7, \mu > 2, \mu \in \mathbb{N}$ описано в [3],[4].

Положим $A(n) \asymp B(n)$, если $\exists c_1, c_2, 0 < c_1 \leqslant c_2$ такие, что $c_1 \cdot$

$B(n) \lesssim A(n) \lesssim c_2 \cdot B(n)$.

Для получения оценок функции Шеннона для классов Поста $D_2$ и $F_i^2, i = 2, 3, 6, 7$ использовались результаты, изложенные в [3]-[6].

**Теорема 1.** *Пусть $\mathcal{K}$ - один из классов $D_2$, $F_i^2, i = 2, 3, 6, 7$. Тогда:*

$$S(\mathcal{K}, n) \asymp \frac{2^n}{n \cdot \sqrt{\log n}}$$

*При этом константы $c_1, c_2$ из определения отношения $\asymp$ равны*

$$c_1 = \sqrt{2/\pi},\ c_2 = \ 2\sqrt{2/\pi}.$$

Отметим, что сложность реализации булевых функций конечными автоматами не коррелирует со сложностью реализации булевых функ-ций полиномами Жегалкина. Сложность полинома Жегалкина булевой функции $f$ определяется как число его ненулевых коэффициентов и обозначается $S^{\oplus}(f)$. В данной работе показано, что сложность полиномов Жегалкина для булевых функций, представляемых сложными автоматами, может кардинально отличаться.

**Теорема 2.** *Существует последовательность $n_p \to \infty$ при $p \to \infty$, такая что для функций $f'_{n_p}, f''_{n_p} \in P_2^{n_p}$ выполнено:*
$S(f'_{n_p}) = S(f''_{n_p}) \sim \frac{2^{n_p+1}}{n_p}$, *но* $S^{\oplus}(f'_{n_p}) \sim \ n_p$ *и* $S^{\oplus}(f''_{n_p}) \sim \ 2^{n_p}$.

Автор выражает благодарность академику Кудрявцеву В.Б. и проф. Бабину Д.Н. за ценные замечания и внимание к работе.

**Литература**

1. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов. —М.: Наука, 1985.

2. Яблонский С. В., Гаврилов Г. П., Кудрявцев В. Б. Функции алгебры логики и классы Поста. —М.: Наука, 1966.

3. Кузьмин А. Д. Реализация функций алгебры логики автоматами, нормальными алгорифмами и машинами Тьюринга // Проблемы кибернетики. —М.: Наука, 1955, Вып. 13, с. 75–96 (РЖМат, 1966, 1В223).

4. Кибкало М. А. Об автоматной сложности некоторых классов булевых функций // Интеллектуальные системы, 2011, т. 14.

5. Коршунов А. Д. О числе монотонных функций // Проблемы кибернетики, 1981, Вып. 38, с. 5–109.

6. Сапоженко А. А. О числе антицепей в многослойных ранжированных множествах // Дискретная математика, 1989, т. 1, вып. 2, с. 110–128.

## ВНУТРЕННЕЕ И ВНЕШНЕЕ НАБЛЮДЕНИЕ КОЛЛЕКТИВА АВТОМАТОВ С ОДНИМ СОСТОЯНИЕМ

**Курганский А.Н. (ИПММ НАН Украины, Донецк)**
*topologia@mail.ru*

В работе рассматриваются коллективы автоматов с одним состоянием, взаимодействующие между собой в однородной вычислительной среде, заданной в виде ориентированного графа. Такие коллективы, а в работе они, чтобы подчеркнуть физические аналогии, названы телами, представляют собой распределенные вычислительные структуры и рассматриваются как цельные автоматоподобные системы. Можно провести аналогию между телами и клеточными автоматами, но вопросы, рассматриваемые в настоящей работе, и их решение делает разницу ними принципиальной. Прежде всего речь идет о том, как мы вводим понятие состояния тела.

В классических примерах автоматов, взаимодействующих со средой, автомат занимает одну вершину среды и в каждый момент времени переходит в одну из соседних вершин среды, при этом скорость изменения состояния автомата равна одному состоянию в единицу времени. Примерами здесь могут служить автоматы в лабиринте, машины Тьюринга, счётчиковые автоматы и др. Особенностью коллективов автоматов является распределенность по среде, которая влечет сложности при определении понятия состояния коллектива. В данной работе развивается подход [2], основывающийся на следующих рассуждениях. Во-первых, по определению элементарной частью коллектива является автомат с одним состоянием, поэтому состояние всего коллектива естественно определять его геометрией, т.е. как инвариант некоторой группы преобразований среды. Во-вторых, всякое вычисление некоторым объектом невозможно без изменений в этом объекте, поэтому вычисление коллективом должно быть связано с изменениями его геометрии. Идея всей работы основывается на следующем примере шахматной доски и нескольких пешек.

Пусть пешки могут делать движение на одну клетку в любом из множества фиксированных на доске направлений в один такт времени. Составим из пешек на доске фигуру, например, букву «О» и посмотрим на пешки как один цельный объект. Определим его скорость перемещения по шахматной доске как среднюю скорость пешек. Пусть объект движется с максимальной скоростью «одна клет-

ка в единицу времени» в одном направлении. Может ли объект при этом каким-либо образом перестроиться из буквы «О», например, в букву «Т»? Нет, в этом примере при максимальной скорости в объекте невозможны вычисления. И, наоборот, в объекте возможны максимальные относительно шахматной доски по скорости вычисления, если он имеет близкую к 0 скорость. Эта идея, почерпнутая в [1], применяется к телам для определения связи между скоростью изменения состояния тела (вычислительное свойство) и скоростью перемещения (динамическое свойство). При этом сравниваются две точки зрения, названные внешним и внутренним наблюдением.

Внешнее наблюдение есть такое определение состояния тела, при котором тело рассматривается относительно среды, вернее, относительно системы отсчета, связанной со средой. В этом случае можно, например, говорить об абсолютных скоростях перемещения.

Внутреннее наблюдение характеризуется тем, что состояние тела определяется относительно самого тела, вернее, относительно системы отсчета связанной с телом. Состояние, определяемое таким образом, называется внутренним. Внутреннее состояние, как следует из определения, не зависит от скорости и деформации среды.

Тело как вычислительная модель определяется с точки зрения внутреннего наблюдения. Но для ее исследования естественно внешнее наблюдение. Например, при внешнем наблюдении естественен вопрос: может ли тело определить свою абсолютную скорость? Для внутреннего наблюдения он не имеет смысла. Еще пример: при внешнем наблюдении рассматриваемые среды являются дискретными и имеют некоторый порядок симметрии вращения. Будет ли среда дискретной и иметь тот же порядок симметрии вращения при внутреннем наблюдении? В целом работа имеет целью именно сравнение внешнего и внутреннего наблюдения.

Назовем $D = \{1, 2, \ldots, m\}$ множеством актуальных пространственных направлений. $(n, m)$-Среда — ориентированный граф. Дугам графа приписывается направление из $D$. Если различные дуги входят в одну и ту же вершину, то говорим, что они пересекаются. Граф вложен в $n$-мерное аффинное метрическое пространство $E$ так, что дуги среды являются отрезками прямых, имеют длину $\frac{1}{n+1}$ и дуги одного направления лежат на параллельных прямых. Зафиксируем систему отсчета в $E$. Пространство $E$ назовем абсолютным, а координаты в нем абсолютными пространственными координата-

ми. Пару $(x,t)$, где $x \in E$, $t$ — время, называем пространственно-временной координатой в абсолютной системе отсчета $O$ или, просто, *событием*. Пусть $D$ является множеством векторов $\{\overrightarrow{1}, \overrightarrow{2}, \ldots, \overrightarrow{m}\}$ в $E$. Обозначим $e_i = (\overrightarrow{i}, 1)$, $1 \le i \le m$. Назовем $\{e_i\}_{1 \le i \le m}$ множеством *актуальных пространственно-временных* направлений, которые образуют абсолютную *актуальную систему отсчета* $Q$.

Элементарным телом или, короче, 1-телом назовем автомат Мили с одним состоянием. Для удобства говорим, что изоморфные 1-тела имеют одинаковые цвета, неизоморфные — разные. Предполагаем, что используются $r$ различных цветов, пронумерованных целыми от 1 до $r$. В каждый момент времени $t \in Z$ 1-тело $b$ находится на какой-либо одной дуге $b(t)$ среды. Входным сигналом тела $b$, находящегося на дуге $e$, входящей в вершину $v$, является упорядоченный набор чисел $(p_{ij})_{1 \le i \le |D|, 1 \le j \le r}$, где $p_{ij}$ — число 1-тел цвета $j$, находящихся на дуге направления $i$, входящей в вершину $v$. Выходом тела $b$ является направление из $D$. Если выходом тела $b$, находящегося в момент $t$ на дуге, входящей в вершину $v$, является направление $i$, то в момент $t+1$ оно находится на дуге направления $i$, исходящей из $v$. Если направления дуг $b(t)$ и $b(t+1)$ совпадают, то говорим, что *внешнее состояние* тела $b$ не изменилось и оно движется прямолинейно. Иначе говорим, что внешнее состояние $b$ изменилось. 1-тело движется прямолинейно, если все пересекающие $b(t)$ дуги пусты.

Обозначим через $\tau_b(t)$ меру изменений внешнего состояния $b$, состоявшихся к моменту времени $t$. По определению, если с $t_1$ до $t_2$ $b$ двигалось прямолинейно, то $\tau_b(t_1) = \tau_b(t_2)$. Обозначим $w_b(t) = \tau_b(t+1) - \tau_b(t)$.

Представим дискретную динамику $b$ на графе непрерывной динамикой в пространстве $E$. Координату $b$ в момент $t$ обозначим через $x_b(t)$. Пусть $b(t) = (v_0, v_1)$, $t \in Z$, и координаты вершин $v_0$ и $v_1$ равны $x_0$ и $x_1$ соответственно, тогда $x_b(t+\lambda) = x_0 + \lambda(x_1 - x_0)$, $0 \le \lambda < 1$.

**Определение.** *Тело — конечное множество 1-тел.*

Тело состоящее из $k$ 1-тел называем также $k$-телом. Если 1-тело принадлежит телу, то называем его элементарной частью этого тела.

Пусть $B = \{b_1, \ldots, b_k\}$ — $k$-тело. Координатой тела $B$ в момент времени $t$ называется $x_B(t) = (x_1(t) + \cdots + x_k(t))/k$. Введем меру $\tau = \tau_B(t)$ изменения внешнего состояния тела $B$, которую также назовем $\tau_B(t)$ собственным временем $B$. Величину $w_B(t) = \tau_B(t+1) - \tau_B(t)$

назовем скоростью собственного времени тела $B$, а величину $v_B(t) = x_B(t+1) - x_B(t)$ абсолютной скоростью перемещения $B$.

**Определение.** *Для любого тела* $B$ $w_B(t) = 0 \Leftrightarrow \forall_{b \in B} w_b(t) = 0$.

Т.е., два тела находятся в среде в одном и том же внешнем состоянии, если одно из них может быть получено из другого прямолинейным сдвигом каждой его элементарной части на равное число шагов в направлении, соответствующем внешнему состоянию.

**Следствие.** *Если* $|v_B(t)| = 1$, *то* $w_B(t) = 0$.

**Следствие.** *Два тела, движущиеся с различной скоростью, находятся в различных внешних состояниях.*

Система отсчета $O_B$ тела $B$ есть такой способ приписывания пространственно-временных координат событиям, при котором, по определению, $x_{BB}(\tau_B) \equiv 0$, $v_{BB}(\tau_B) \equiv 0$ и $w_{BB}(\tau_B) \equiv 1$, где $x_{AB}(\tau_B)$, $v_{AB}(\tau_B)$, $w_{AB}(\tau_B)$ и $\tau_{AB}(\tau_B)$ обозначают координату, скорость перемещения, скорость собственного времени и собственное время тела $A$ в момент времени $\tau_B$ в системе отсчета $O_B$ соответственно.

**Определение.** *Тела* $A$ *и* $B$ *находятся в одном внутреннем состоянии в моменты собственного времени* $\tau_A$ *и* $\tau_B$ *соответственно, если* $\{(b, x_{bA}(\tau_A)) | b \in A\} = \{(\varphi(b), x_{bB}(\tau_B)) | b \in B\}$ *для некоторой биекции* $\varphi : A \to B$ *такой, что* $b \in A$ *и* $\varphi(b) \in B$ *изоморфны.*

Состояние тела как вычислительной модели есть его внутреннее состояние. Тело $B$ инерциальное в $O_A$, если $v_{BA}$ и $w_{BA}$ константы. Система отсчета $O_A$ инерциальная, если $A$ инерциальное в $O$.

**Определение.** *Среда корректная, если любые инерциальные системы отсчета в ней можно связать аффинным преобразованием.*

**Теорема.** $\{e_i\}_{1 \le i \le m}$ *— собственные вектора аффинных преобразований, связывающих инерциальные системы отсчета.*

**Следствие.** *Верно:* $v_{AB} = -v_{BA}$, $w_{AB} \cdot w_{BA} = 1 - v_{AB}^2 = 1 - v_{BA}^2$.

**Следствие.** *Если* $m \neq n+1$, *то* $(n, m)$*-среда не корректная.*

**Теорема.** *Степень симметрии вращения среды может не совпадать при внутреннем и внешнем наблюдении.*

Абсолютная скорость и актуальные пространственные направления не имеют смысла при внутреннем наблюдении.

**Литература**

1. Пуанкаре А. О науке. – М.: Наука, 1983. – 560 с.

2. Kurganskyy O. A state of a dynamic computational structure distributed in an environment: a model and its corollaries // Труды

ИПММ НАНУ, 2010, вып. 21, с. 150-160

# О МИНИМИЗАЦИИ МОНОФУНКЦИОНАЛЬНЫХ КЛАССОВ БИНАРНЫХ КЛЕТОЧНЫХ АВТОМАТОВ С НЕРАЗРЕШИМЫМ СВОЙСТВОМ ОБРАТИМОСТИ

**Кучеренко И.В. (МГУ им. М. В. Ломоносова)**
*kucherenko@intsys.msu.ru*

Клеточные автоматы (КА) являются дискретной математической моделью процессов, для которых существенна не только временная, но и пространственная протяженность [1]. Важное семейство клеточных автоматов образуют обратимые КА, то есть такие, в которых "предыстория" возникновения конфигурации определяется однозначно. Класс обратимых КА представляет как теоретический, так и прикладной интерес — в связи с задачами защиты информации, синтеза квантовых вычислителей, проектирования вычислительных систем с пониженным энергопотреблением и других.

В работе пойдет речь о задаче алгоритмического распознавания свойства обратимости в классах двумерных бинарных КА (у которых ячейка имеет два состояния). Автором установлено, что свойство обратимости не распознаваемо в классе всех двумерных бинарных клеточных автоматов [2]. С другой стороны, в классе двумерных бинарных клеточных автоматов, в которых содержатся только КА с линейными локальными функциями переходов, свойство обратимости разрешимо [3]. В связи с этим возникает задача классификации "естественных" классов КА на те, в которых свойство обратимости разрешимо, и те, для которых это не так.

В работе рассматриваются классы бинарных двумерных КА, имеющих фиксированную локальную функцию переходов (в таком классе варьируются исключительно вектора в локальном шаблоне соседства); такие классы будем называть монофункциональными. Автором построен монофункциональный класс двухмерных бинарных клеточных автоматов, в котором задача распознавания свойства обратимости является алгоритмически не разрешимой. Получена оценка для числа существенных переменных локальной функции переходов в данном классе.

Приведем необходимые для понимания полученного результата определения. Формально клеточный автомат $\sigma$ представляет из себя четверку вида $(Z^k, E_n, V, \varphi)$, где $Z^k$ — совокупность всех $k$-мерных векторов с целочисленными координатами, $E_n$ — конечное множество из $n$ элементов, природа которых не существенна. Для просто-

ты их можно считать числами из множества $\{0, 1, \ldots, n-1\}$. $V = \{v_1, v_2, \ldots, v_m\}$ — упорядоченный набор различных ненулевых векторов из $Z^k$. $\varphi : (E_n)^{m+1} \mapsto E_n$, $\varphi(0, 0, \ldots, 0) = 0$. Элементы множества $Z^k$ называются ячейками, $E_n$ — состояниями ячеек, 0 — состояние покоя. При помощи шаблона соседства $V$ каждой ячейке $\alpha$ ставится в соответствие набор векторов $V(\alpha) = \{\alpha, \alpha+v_1, \alpha+v_2, \ldots, \alpha+v_m\}$, который называется ее окрестностью. Функция $\varphi$ называется локальной функцией переходов клеточного автомата.

Функции $g : Z^k \mapsto E_n$ называются состояниями КА. Основная функция переходов $\Phi$ задается как отображение множества всех состояний клеточного автомата $\sigma$ в себя, причем если $g = \Phi(g')$, то $g(\alpha) = \varphi(g'(\alpha), g'(\alpha + v_1), g'(\alpha + v_2), \ldots, g'(\alpha + v_m)), \forall\alpha$. Функционирование КА определяется как последовательность его состояний $g_0, g_1, g_2, \ldots$, получающаяся в результате применения основной функции переходов к некоторому его состоянию $g_0$, то есть $g_t = \Phi(g_{t-1}) = \Phi^t(g_0)$, $t$ — натуральное число. Состояние клеточного автомата, в котором только конечное число ячеек находится в ненулевом состоянии, называется конфигурацией.

Клеточный автомат, основная функция переходов которого инъективна на множестве всех конфигураций, называется обратимым. По теореме Мура-Майхилла [1] множество обратимых клеточных автоматов совпадает с множеством КА, основная функция переходов которых является сюръективной.

Пусть $\varphi$ — булева функция, зависящая от $m + 1$ переменных и сохраняющая ноль. Множество двумерных клеточных автоматов с локальной функцией переходов $\varphi$ обозначим через $CA(2, 2, m, \varphi)$. Будем задавать индивидуальные клеточные автоматы из множества $CA(2, 2, m, \varphi)$ набором из $m$ двумерных ненулевых целочисленных векторов $V$ (их шаблоном соседства). Задача алгоритмического распознавания свойства обратимости заключается в построении машины Тьюринга (МТ), которая на наборе $V = ((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m))$, записанному на ее ленте в виде последовательности из $2 \cdot m$ натуральных чисел $x_1, y_1, x_2, y_2, \ldots, x_m, y_m$ в унитарной записи (отдельные числа разделяются одиночной буквой “0”; в начальном состоянии на всей “свободной” части ленты записана буква “0”, головка находится над самой левой буквой “1” конфигурации), останавливалась, при этом в ячейке ленты, находящейся под головкой в момент остановки, должно находиться буква “1”, если клеточный

автомат $\sigma = (Z^2, E_2, V, \varphi)$ обратим, или "0", если $\sigma$ не обратим.

Основной результат работы получается сведением проблемы обратимости КА к проблеме остановки МТ в специальной формулировке. Применяемая конструкция позволяет дополнительно получить оценку на число переменных функции $\varphi$, но это требует определения параметров исходной проблемы. Обозначим через $q$ число состояний головки машины Тьюринга $\varkappa$, обладающей следующими свойствами.

1. МТ $\varkappa$ имеет одну ленту с двумя состояниями.

2. Рассматриваются только вычисления $\varkappa$, в которых в начальный момент времени головка стоит над самой левой буквой слова.

3. Проблема остановки $\varkappa$ на словах, представляющих из себя наборы из одних единиц, не разрешима.

**Теорема 1.** *Существует булева функция $\varphi(x_0, x_1, \ldots, x_m)$, такая, что в классе $CA(2, 2, m, \varphi)$ свойство обратимости алгоритмически не разрешимо, при этом*

$$m \leq 7 \cdot (5 + 2 \cdot ]\log_2(26 \cdot (q + 5))[) - 1.$$

Автор выражает благодарность своему научному руководителю академику В. Б. Кудрявцеву за постановку задачи и внимание к работе.

**Список литературы**

1. Кудрявцев В. Б., Подколзин А. С., Болотов А. А. Основы теории однородных структур — М.: Наука, 1990.

2. Кучеренко И. В. О разрешимости обратимости клеточных автоматов // Интеллектуальные системы. — 2004. — Т. 8, вып. 1-4. — С. 465–482.

3. Кучеренко И. В. О структуризации класса обратимых бинарных клеточных автоматов // Интеллектуальные системы. — 2005. — Т. 9, вып. 1-4. — С. 445–456.

# О ВЫРАЗИМОСТИ СУПЕРПОЗИЦИЯМИ ГРУППОВЫХ АВТОМАТОВ МЕДВЕДЕВА

**Летуновский А.А.(г.Москва)**
*alekseyletunovskiy@gmail.com*

Рассматривается задача выразимости конечного автомата $A$ суперпозициями систем вида $\Phi \cup \nu$, где $\Phi$ состоит из всех булевых функций и "задержки", $\nu$ - произвольная конечная система автоматов. Ранее автор показал, что для автомата $A$ с безусловными переходами существует алгоритм проверки $A \in [\Phi \cup \nu]$. В настоящей работе показано, что задача выразимости через системы $\Phi \cup \nu$ групповых автоматов Медведева алгоритмически разрешима.

Задача выразимости автоматов относительно суперпозиции наталкивается на существенные трудности[2]. В общем случае она является алгоритмически неразрешимой, а разрешимые случаи сводятся к теоретико-групповым конструкциям[3,4]. Относительно суперпозиции не существует конечных полных систем, а как показал Бабин Д.Н.[5] арность полных систем может быть выбрана равной 2. Задача полноты относительно композиции разрешима, когда в базисе всегда есть булевы функции[6,7]. В нашем случае предполагается наличие в выражающей системе автоматов штрих Шеффера и "задержка". Ранее автором было показано существование алгоритма выразимости константных автоматов[8], а в настоящей работе в список автоматов, для которых есть алгоритм выразимости через системы с добавкой из штриха Шеффера и "задержки", включены групповые автоматы Медведева.

Пусть $E_2 = \{0, 1\}$, функции вида $g : E_2^n \to E_2$ называются булевыми функциями, их множество обозначается через $P_2$. Пусть $E_2^\infty$-множество всех сверхслов вида $a(1)a(2)...$, где $a(j) \in E_2$, $j = 1, 2, ....$ Пусть

$$f : (E_2^\infty)^n \to (E_2^\infty)^m$$

- автоматная функция( $a$-функция), т.е. она задается рекуррентно соотношениями, согласно каноническим уравнениям [1].

Шестерка

$$(E_2^n, E_2^s, E_2^m, \phi, \psi, q_0)$$

, где вектор $q = (q_1, ..., q_s) \in E_2^s$ состояние $a$-функции $f$, $q_0$ ее начальное состояние, буквы $a = (a_1, a_2..., a_n) \in E_2^n$ и $b = (b_1, ..., b_m) \in E_2^m$ входная и выходная буквами, а сверхслова $a(1)a(2)...$ и $b(1)b(2)...$ - входные и выходные сверхслова соответственно, вектор-функции $\phi : E_2^n \times E_2^s \to E_2^s$ и $\psi : E_2^n \times E_2^s \to E_2^m$ функции переходов и выходов соответственно, называется автоматом, задающим автоматную функцию.

Класс всех $a$-функций обозначим через $P$. Автомат $M$ называется *автоматом Медведева,* если $s = m, \psi(a, q) = q$.

В этом классе обычным образом введем операции суперпозиции [1].

Пусть $M \subseteq P$, обозначим через $[M]$ - множество $a$-функций, получающихся из $M$ с помощью операций суперпозиции.

Автоматную функцию $G_0$, задаваемую уравнениями

$$\begin{cases} q(1) = 0, \\ q(t+1) = a(t), \\ b(t) = q(t), \end{cases}$$

назовём автоматной функцией "задержки".

Если для каждого $a \in A$ $\phi_a(q) : E_2^s \to E_2^s$, где $\phi(q, a) = \phi_a(q)$ является биекцией, то автомат называется групповым.

Мы будем рассматривать задачу выразимости групповых автоматов Медведева через системы вида $\Phi \cup \nu$, где $\Phi$ - состоит их всех булевых функций и "задержки", $\nu$-произвольная конечная система автоматов.

**Теорема.** *Пусть $A$ - произвольный групповой автомат Медведева , тогда задача $A \in [\Phi \cup \nu]$ является алгоритмически разрешимой.*

**Список литературы**

1. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов, Наука, М., 1985.

2. Кратко М. И. Алгоритмическая неразрешимость проблемы распознавания полноты для конечных автоматов // ДАН СССР, 1964, т.155, N 1, с.35–37.

3. М. Арбиб Алгебраическая теория автоматов языков и полугрупп, "Статистика М.,1975.

4. Алешин С. В. Об одном следствии теоремы Крона–Роудза, Дискретная математика, том 11, вып. 4, 1999 год, стр. 101-109

5. Бабин Д. Н. О полноте двухместных автоматных функций относительно суперпозиции, Дискретная матетматика, том 1, выпуск 4, 1989, стр. 423-431

6. Буевич В. А. Условия А-полноты для автоматов, изд. МГУ, 1986 г.

7. Бабин Д. Н. О классификации автоматных базисов Поста по разрешимости свойств полноты и А-полноты, ДОКЛАДЫ АКАДЕМИИ НАУК, N 4, Т.367, 1999 с. 439-441

8. Летуновский А. А. О выразимости константных автоматов, Интеллектуальные системы , том 9, вып. 1-4, 2005 год с. 457-469

# ЧАСТИЧНОЕ УГАДЫВАНИЕ СВЕРХСОБЫТИЙ, ОБРАЗОВАННЫХ ДЕТЕРМИНИРОВАННЫМИ КОНТЕКСНО-СВОБОДНЫМИ ЯЗЫКАМИ

**Мастихина А.А. (МГУ им.М.В.Ломоносова)**
*anmast@yandex.ru*

Рассматривается задача частичного угадывания любой последовательности из нулей и единиц из заданного множества детерминированными, но, возможно, бесконечными автоматами. Детерминированность автомата означает, что после подачи на его вход $t$ первых символов сверхслова он однозначно выдает некоторый выходной символ. Ранее критерии частичной угадываемости были получены для общерегулярных сверхсобытий [3] и для сверхсобытий вида $L^\infty$, где $L$ порождается простой $LL(1)$-грамматикой [4].

Выходное сверхслово некоторого автомата $\mathfrak{A}$ при подаче на его вход сверхслова $\alpha \in \{0,1\}^\infty$ будем обозначать через $y_\alpha^{\mathfrak{A}}$.

Автомат $\mathfrak{A}$ *угадывает сверхслово* $\alpha \in \{0,1\}^\infty$ *со степенью* $\sigma \in [0,1]$, если

$$c^{\mathfrak{A}}(\alpha) = \underline{\lim}_{t\to\infty} \frac{1}{t} \sum_{i=1}^{t} (1 - |y_\alpha^{\mathfrak{A}}(i) - \alpha(i+1)|) = \sigma.$$

Автомат $\mathfrak{A}$ *частично угадывает* множество сверхслов $A$, если для любого $\alpha \in A$ найдется такое $\sigma > 0$, что выполнено $c^{\mathfrak{A}}(\alpha) > \sigma > 0$. Множество $A$ *частично угадываемо*, если найдется частично угадывающий его автомат $\mathfrak{A}$.

*Степень угадывания* множества $A$

$$c^{\mathfrak{A}}(A) = \inf_{\alpha\in A} c^{\mathfrak{A}}(\alpha).$$

Определим *автомат с магазинной памятью* как семерку $(Q, \Sigma, \Gamma, Z, \Phi, q_0, F)$, где $Q$ — множество состояний, выделенное начальное состояние $q_0 \in Q$ и некоторое множество $F \subseteq Q$ заключительных состояний, $\Sigma$ — входной алфавит, $\Gamma$ — магазинный алфавит, $\Phi$ — множество команд, каждая команда — отображение $(Q \times (\Sigma \cup \lambda) \times \Gamma \to Q \times \Gamma^*)$, где $\lambda$ — пустое слово, $Z$ — символ для обозначения пустого магазина. Тактом работы будем считать выполнение возможных команд после поступления одного входного символа. Выполнение команды, содержащей в левой части $\lambda$, не будем считать отдельным тактом.

*Конфигурацией* будем называть пару $(q, \xi), q \in Q, \xi \in \Gamma^*$: текущее состояние и содержимое магазина.

МП-автомат *допускает язык* $L$, если только после подачи слов $\alpha \in L$ на начальную конфигурацию $(q_0, S)$ автомат приходит в конфигурацию $(q, Z), q \in F$.

Автомат называется *детерминированным*, если для каждой конфигурации $(q, \xi)$ имеется либо не более одной команды вида $(q, a, \xi]_1 \to r, \beta), q, r \in Q, \xi, \beta \in \Gamma^*$ для каждого $a \in \Sigma$ и ни одной команды вида $(q, \lambda, \xi]_1 \to r', \beta'), r' \in Q, \beta' \in \Gamma^*$, либо ни одной команды вида $(q, a, \xi]_1 \to r, \beta)$ и не более одной вида $(q, \lambda, \xi]_1 \to r', \beta')$.

Класс языков, допускаемых детерминированными автоматами с магазинной памятью — класс детерминированных контекстно - свободных языков.

Так как класс детерминированных контекстно-свободных языков не замкнут относительно конкатенации, будем рассматривать такие языки, что их итерация есть детерминированный контекстно-свободный язык.

Далее будем рассматривать $Z$ как элемент $\Gamma$, а $(q_0, Z)$ возьмем в качестве начальной конфигурации, то есть автомат работает с пустым магазином.

Строится допускающий детерминированный автомат с магазинной памятью для итерации языка $L$. На него подается его сверхитерация. Можно заметить, что при подаче любого сверхслова из множества $L^\infty$ автомат оказывается в какой-нибудь конфигурации $(q, Z), q \in F$ бесконечное число раз. Это может быть верно и для других сверхслов, но предполагается, что сверхслова не из $L^\infty$ просто не поступают на вход.

**Теорема.** *Пусть язык $L^*$ допускается детерминированным автоматом с магазинной памятью $\mathfrak{A} = (Q, \{0, 1\}, \Gamma, Z, \Phi, q_0, F)$. Множество $L^\infty$ является частично угадываемым тогда и только тогда, когда для некоторой пары $(q, A), q \in Q, A \in \Gamma$ существует только одна команда вида $(q, a, A \to r, \beta), a \in \{0, 1\}, r \in Q, \beta \in \Gamma^*$.*

Необходимость обосновывается тем, что в противном случае в любой конфигурации $\mathfrak{A}$ есть два варианта следующей входной буквы, поэтому для любого детерминированного автомата возможно построение сверхслова из $L^\infty$ с любым количеством неугаданных подряд символов. Для такого сверхслова можно выбрать подпоследовательность, на которой доля угаданных символов стремится к ну-

лю. Поэтому степень угадывания соответствующего множества будет равна нулю.

Если же есть конфигурации, в которых возможна только одна команда (не с $\lambda$ в левой части), то следующая буква известна. Для остальных конфигураций выбирается та буква, которая отдаляет допускающий автомат от конфигурации с одной альтернативой. С этой целью для состояния и магазинного символа вычисляется кратчайшее слово, приводящий автомат в конфигурацию с одной альтернативой без укорачивания магазина, если такой существует, либо кратчайший путь, укорачивающий магазин.

Для частичного угадывания используется автомат с магазинной памятью с выходом.

**Пример.**

Рассмотрим детерминированный автомат с магазинной памятью $\mathfrak{A} = (Q, \{0,1\}, \Gamma, Z, \Phi, q_0, F)$, где $Q = \{q_0, q_1, q_2\}, \Gamma = \{A, Z\}, F = q_0$, а множество команд $\Phi$ таково:

$$
\begin{array}{l}
(q_0, 1, Z \to q_1, \lambda), \\
(q_1, 1, Z \to q_1, A) \\
(q_1, 0, A \to q_1, AA), \\
(q_1, 1, A \to q_2, A), \\
(q_2, \lambda, Z \to q_0, \lambda), \\
(q_2, 0, A \to q_2, \lambda), \\
(q_2, 1, A \to q_1, A).
\end{array}
$$

Данный автомат допускает итерацию языка
$L = \{110^{n_1}10^{n_2}1...10^{n_{2k}} | \Sigma_{i=1}^{k} n_{2i} = \Sigma_{i=1}^{k} n_{2i-1} + 1\}$.

Добавим автомату выходную функцию, сопоставив парам $(q, B)$, $q \in Q$, $B \in \Gamma$, для которых есть команды, начинающиеся на $q, a, B$, $a \in \{0,1\}$, значение $f(q, B) \in \{0,1\}$. Полученное устройство $\mathfrak{A}'$ будет частично угадывать $L^{\infty}$.

В конфигурациях $(q_1, Z)$ и $(q_0, Z)$ есть только по одной команде для входной буквы 1, поэтому выходная функция будет $f(q_1, Z) = 1, f(q_0, Z) = 1$, и в этих конфигурациях автомат будет угадывать.

Введем функцию $\mu : Q \times \Gamma \to \{0, 1, 2, ...\}$, равную наименьшей длине входного слова, переводящего автомат из конфигурации $(q, A\xi)$ в $(r, \xi)$ для некоторого $r \in Q$, то есть укорачивающего магазин.

$\mu(q_2, A) = 1$, причем укорачивание происходит при выполнении команды $(q_2, 0, A \to q_2, \lambda)$, поэтому зададим $f(q_2, A) = 1$.

$\mu(q_1, A) = 2$, так как после команды $(q_1, 1, A \to q_2, A)$ автомат попадает в конфигурацию $(q_2, A\xi)$, слово, укорачивающее магазин — 10, поэтому $f(q_1, A) = 0$.

Степень угадывания $c^{\mathfrak{A}'}(L^\infty) = \frac{1}{2}$, и $\frac{1}{2}$ достигается на сверхслове $(1110)^\infty$.

Автор выражает благодарность профессору Э.Э.Гасанову за постановку задачи и помощь в работе.

**Литература**

1. Вереникин А. Г., Гасанов Э. Э. Об автоматной детерминизации множеств сверхслов // Дискретная математика. — 2006. — Т.18, №2. — С. 84–97.

2. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции: Пер. с англ. – М.: Мир, 1978

3. Мастихина А. А. Критерий частичного предвосхищения общерегулярных свехсобытий // Дискретная математика — 2011.

4. Мастихина А. А. Частичное угадывание некоторых контекстно-свободных языков // Материалы XVIII Международной конференции студентов, аспирантов и молодых учёных «Ломоносов».

## ВТОРАЯ АВТОМАТНАЯ ФУНКЦИЯ И С НЕЮ СВЯЗАННЫЕ КЛАССЫ РЕГУЛЯРНЫХ ЯЗЫКОВ

**Пархоменко Д.В. (МГУ им. М. В. Ломоносова)**
*dcdenis@rambler.ru*

В докладе будут рассмотрены множества слов конечного алфавита, возникающие на выходе детерминированных автоматов и изучены их свойства. Будут введены ранее не исследовавшиеся классы регулярных языков. Введено новое понятие второй автоматной функции.

*Пусть задан КДА* $V = (A, Q, B, \varphi, \psi, q_0)$, $|A| = |B|$, *и его автоматная функция* $f_V \colon A^* \to B^*$. *Тогда функция* $\kappa \colon B^* \to \mathbb{N} \cup \{0\}$, $\kappa(\beta) = |\{\alpha \in B^* \mid f_V(\alpha) = \beta\}|$ *называется второй автоматной функцией автомата* $V$.

Пусть для любого натурального $p : L_p(V) = \{\beta \in B^* | \kappa_V(\beta) \in p\}$. В частности, при $p = 1, L_p(V)$ суть автоматно перечислимое множество слов. Очевидно, для любого автомата $V$ : $L_p(V) \subseteq L_{p-1}(V)$, для всех $p \geq 2$. Справедлива:

**Теорема 1.** *Для любого конечного инициального автомата* $V$ *и для всякого* $p \geq 1$, $L_p(V)$ *- регулярный язык.*

**Утверждение.** *Пусть дан автомат* $V$. *Если для некоторого* $i \in \aleph$, *слово* $\beta \in L_p(V)$, *то найдется буква* $b$ *выходного алфавита автомата* $V$ *такая, что* $\beta b \in L_p(V)$.

**Следствие.** *Для любого натурального* $p$ *и автомата* $V, L_p(V)$ *либо бесконечное множество, либо пустое.*

Пусть $\mathcal{L}_p = \{L_p(V)|V\}$. Имеет место

**Теорема 2.** *Для любых натуральных* $i<j$ *выполнено:* $\mathcal{L}_i \nsubseteq \mathcal{L}_j$.

Автор выражает глубокую благодарность своему профессору, д.ф.м.н. Бабину Дмитрию Николаевичу за постановку задачи и ценные советы в процессы работы.

### Литература

1. Кудрявцев В. Б., Алешин С. В., Подколзин А. С., *Элементы теории автоматов*, Изд-во МГУ, 320 с, 1985.

# КРИТЕРИЙ ПОЛНОТЫ НЕКОТОРЫХ БЕСКОНЕЧНЫХ СИСТЕМ ВО МНОЖЕСТВЕ АВТОМАТНЫХ ОТОБРАЖЕНИЙ

**Родин А.А. (МГУ имени М.В.Ломоносова)**
*tyman307@rambler.ru*

Через $P^2$ обозначим множество всех ограниченно-детерминированных функций (автоматных отображений), входные и выходные переменные которых принимают значения из множества бесконечных последовательностей, составленных из нулей и единиц.

Будем считать, что на множестве $P^2$ определена операция суперпозиции[1]. Пусть $M \subseteq P^2$. Замыкание множества $M$ относительно суперпозиции обозначим через $[M]$. Множество $M \subseteq P^2$ называется полным, если $[M] = P^2$.

Пусть $D$ - произвольный замкнутый класс алгебры-логики. Обозначим через $P_D^2$ множество ограниченно-детерминированных функций, в каждом состоянии которых реализуется функция алгебры-логики, принадлежащая $D$. Например, если $D = \{0, 1\}$, то $P_D^2$ - множество автоматов Мура. Если $D = \{x\}$, будем обозначать $P_D^2$ через $P_0$.

Пусть $\Omega^D$ - совокупность всех подмножеств $M$ множества $P^2$, содержащих $P_D^2$ и таких, что множество $M \setminus P_D^2$ конечно.

Известно[3], что в $P_2$ существует континуум предполных классов, содержащих $P_D^2$. Вместе с тем, для некоторых классов $D$ существует эффективный критерий распознавания полноты систем из $\Omega^D$:

**Теорема 1.** *Пусть $D$ содержит тождественную функцию а.-л. и одну из констант. Существует эффективный критерий для распознавания полноты множеств, принадлежащих совокупности $\Omega^D$.*

Ясно, что формулировка этого критерия будет зависеть от множества $D$.

Отметим, что эффективный критерий был получен Алешиным С.В. для случая, когда $D = \{0, 1, x, \bar{x}\}$ [6].

С другой стороны, из [1] следует, что если $D \in \{0, 1\}$, то эффективного критерия распознавания полноты не существует.

Таким образом, пока неисследованными остаются случаи, когда $x \in D, 0, 1 \notin D$. Несложно видеть, что если эффективный критерий существует для случая $D_0 = \{x\}$, то существует и для всех $D \supseteq D_0$.

Для случая $D_0 = \{x\}$ имеют место следующие утверждения.

**Теорема 2.** *Пусть $\Omega(C)$ - совокупность всех подмножеств $M$ множества $P^2$, содержащих $P_0$ и константную о.-д. функцию $C$, таких, что множество $M \setminus P_0$ конечно. Для каждой $C$ существует эффективный критерий распознавания полноты систем из $\Omega(C)$.*

Пусть о.-д. функции $G_0(x), G_1(x)$ - нулевая и единичная задержка соответственно[1]. Обозначим $G = [\{G_0(x), G_1(x)\}]$

**Теорема 3.** *Пусть $g(x) \in G, \Omega(g)$ - совокупность всех подмножеств $M$ множества $P^2$, обладающих следующими свойствами:*

*1) $M$ содержит $P_0 \cup \{g\}$*

*2) Множество $M \setminus P^0$ конечно.*

*Существует критерий распознавания полноты систем из $\Omega(g)$.*

В заключение автор выражает благодарность своему научному руководителю Буевичу В.А.

**Литература**

1.Кудрявцев В.Б., Алешин С.В., Подколзин А.С "Введение в теорию автоматов"Наука, Москва, 1985.

2.Кудрявцев В.Б. "О мощности множеств предполных множеств некоторой функциональной системы, связанной с автоматами"Сборник "Проблемы кибернетики"выпуск 13 М. Физматгис, 1965.

3.Родин А.А. "О предполных классах во множестве автоматных отображений"Материалы конференции "Современные проблемы математики и их приложений"2009 г., стр. 372

4.Буевич В.А. "Критерий полноты систем, содержащих все одноместные ограниченно-детерминированные функции"Дискретная математика 2000, 12, выпуск 4.

5.Яблонский С.В., Гаврилов Г.П., Кудрявцев В.Б. "Функции алгебры логики и классы Поста"Наука, Москва, 1965

6.Алешин С.В. "Über ein Vollstänig klits kriterium für Automaten abbildungen beruglich der Superposition"Rostoker Math. Kolloq. 1977. No.5. s.119-132

## ЛИНЕЙНО РЕАЛИЗУЕМЫЕ ПЕРЕХОДНЫЕ СИСТЕМЫ.

**Родин С.Б. (Московский государственный университет имени М. В. Ломоносова)**
*sergei_rodin@mail.ru*

На практике часто необходимо решать задачу перехода от автоматного описания функционирования на язык схем. Например, при логическом синтезе чипов на первом этапе функционирование чипа описывается как конечный автомат. Переход к описанию на языке схем осуществляется с помощью кодирования алфавита состояний, входного алфавита и выходного алфавита в алфавите $\{0, 1\}$.

При этом важно выбрать кодирование, при котором достигается возможно меньшая сложность схемы.

В работе изучаются сложность реализации автоматов специального вида, т.н. автоматы без выхода или переходные системы. С формальной точки зрения переходная система это тройка $V = (A, Q, \varphi)$, где $A$-входной алфавит, $Q$-алфавит состояний, $\varphi$ функция, которая по текущему входу и состоянию определяет состояние переходной системы в следующий момент времени. Кодирование алфавита состояния - это отображение алфавита $Q$ в ${E_2}^k$, при котором каждому состоянию из $Q$ ставится в соотвествие вектор из ${E_2}^k$. Соотвественно кодирование входного алфавита - это отображение алфавита $A$ в ${E_2}^p$, при котором каждому элементу из $A$ ставится в соотвествие вектор из ${E_2}^p$. При этом функции перехода $\varphi$ преобразуется в булевский оператор $\phi : {E_2}^{p+k} \to {E_2}^k$, где $p$-длина кодового набора символов множества $A$, $k$-длина кодового набора символов множества $Q$. Данный оператор может быть рассмотрен как набор $k$ булевских функций от $n+k$ переменных. А его сложность определить как максимальную сложность получаемых булевских функций. Как известно каждой булевской функции единственным образом соответствует полином Жегалкина. Мы будем понимать сложность оператора как максимальную из сложностей полиномов Жегалкина функций, задающих этот оператор, т.е. максимальная степень полиномов. Таким образом, установив связь между переходной системой, кодировкой и возникающим полиномом, можно установить минимальную сложность реализации переходной системы.

Естественно начинать такого рода исследования с „простейших“, линейных полиномов. Основной задачей данной работы было изу-

чение переходных систем, у которых существует кодирование, такое что получаемые при данном кодировании, булевские функции являются линейными.

Вообще говоря, возникаемый при кодировании оператор, является частично определенным, так как значение оператора определенно только на кодирующих наборах. Доопределение может как „упростить“ оператор, так и „усложнить“ его. В работе будут изучаться переходные системы мощность множества состояний есть степень 2. Зная какие переходные системы имеют линейную реализацию, можно доопределять частично определенные операторы до линейных или установить что это невозможно.

Введем некоторые понятия.

**Определение 1:** Нумерованной переходной системой назовем тройку $(A, Q, \varphi)$, где $A$-входной алфавит, $Q = \{0...n-1\}$, $\varphi$ - функция переходов. В работе изучаются нумерованные переходные системы с входным алфавитом $A = E_2$ и числом состояний $n = 2^k$.

**Определение 2:** Кодированием множества $Q = \{0...n-1\}$ назовем взаимоднозначное отображение $F : \{0...n-1\} \to {E_2}^k$.

Каждое кодирование $F$ для переходной системы на множестве $Q$ порождает булевский оператор $\phi : {E_2}^{k+1} \to {E_2}^k$, где

$$\phi(a, \alpha_1, ..., \alpha_k) = F(\varphi(a, F^{-1}(\alpha_1, ..., \alpha_k))), a \in A, \alpha_i \in E_2.$$

Данный оператор может быть рассмотрен как набор $k$ булевских функций, зависящих от $k+1$ переменной. Обозначим этот набор через $\mathcal{F}_V(F)$.

**Определение 3:** Если для заданной нумерованной переходной системы $V$ существует кодирование $F$, такое что все элементы $\mathcal{F}_V(F)$ являются линейными функциями алгебры логики, назовем такую переходную систему линейно реализуемой посредством кодирования $F$ или просто линейно реализуемой.

Выделим из всех кодирований "стандартное"кодирование.

**Определение 4:** Кодирование $F_0 : \{0...n-1\} \to {E_2}^k$ назовем стандартным, если код элемента есть его двоичное представление.

Каждому кодированию $F$ можно сопоставить перестановку $s_F$ на множестве $Q : \{0...n-1\}$ по правилу $s_F(i) = {F_0}^{-1}(F(i))$.

Пусть задана переходная система $V = (E_2, Q, \varphi)$ и некоторое кодирование $F$. Рассмотрим перестановку $s_F$. Обозначим через $V_{s_F}$ переходную систему, с входным алфавитом $E_2$, алфавитом состояний

$Q = \{0...n-1\}$ и функцией $\varphi$, такой что $\varphi(0,q) = s_F(p_0({s_F}^{-1}(q)))$ для любого $q \in Q$, $\varphi(1,q) = s_F(p_1({s_F}^{-1}(q)))$ для любого $q \in Q$.

**Теорема 1:** *Пусть задана переходная система $V = (E_2, Q, \varphi)$ и некоторое кодирование $F$. Тогда булевский оператор $\phi_V$, порождаемый кодированием $F$ равен булевскому оператору, порождаемому кодированием $F_0$ переходной системы $V_{s_F}$.*

Пусть задана переходная система $V = (E_2, Q, \varphi)$ и некоторое кодирование $F$. Обозначим через $\phi_V : {E_2}^{k+1} \to {E_2}^k$, булевский оператор, порождаемый кодированием $F$ переходной системы $V$.
Обозначим через $X_V = \{s : Q \to Q | \exists a \in E_2, s(q) = \varphi(a,q)i$ для $\forall q\}$ А через $S_V =< X_V >$, замыкание множества $X_V$ относительно операции умножения подстановок.[4]
**Определение 5:** $S_V$ назовем внутренней полугруппой переходной системы $V$. $X_V$ порождающее множество внутреней полугруппы.
Поскольку входной алфавит $E_2$, то множество $X_V$ состоит из двух элементов. Обозначим через $p_0$ подстановку, соответствующую входному символу 0, через $p_1$ подстановку, соответствующую входному символу 1.
Поскольку $n = 2^k$, то подстановки на множестве $Q = \{0...n-1\}$ могут быть представлены как многочлены над полем Галуа $F_{2^k}$ [1].
Обозначим через $P_n$ множество подстановок на множестве $E_n$.
Обозначим через $H_+$ перестановки, соответствующие многочленам $x + c$ над полем Галуа $F_n$, где $c \in E_n$ - константа.
Порядок умножения подстановок слева направо. То есть если заданы перестановки $p_1$ и $p_2$, то значение их произведения на элементе $i$ есть $(p_1 \cdot p_2)(i) = p_2(p_1(i))$.

**Теорема 2:** *Пусть задана переходная система $V = (E_2, Q, \varphi)$. $V$ линейно реализуема посредством кодирования $F$, тогда и только тогда, когда существует подстановка $c$ такая, что $p_0 = c \cdot h_1$, $p_1 = c \cdot h_2$, где $h_1, h_2 \in {s_F}^{-1} \cdot H_+ \cdot s_F$.*

Обозначим через $V(n)$ множество нумерованных переходных систем со входным алфавитом $E_2$ и алфавитом состояний
$Q = \{0...n-1\}$.

**Теорема 3:** *Мощность множества $V(n)$ равна $n^{2n}$.*

**Теорема 4:** *Число различных нумерованных переходных систем с $n$ состояниями, линейно реализуемых посредством стандартного кодирования $F_0$, равно $n^{\log_2(n)+2}$.*

**Теорема 5:** *Число различных линейно реализуемых нумеровaн-*

*ных переходных систем с $n$ состояниями не превосходит* $n^{\log_2(n)+2} \cdot (n-2)!$.

**Теорема 6:** *Число линейно реализуемых переходных систем с $n$ состояниями есть $o(V(n))$.*

В заключение, автор выражет благодарность Алёшину Станиславу Владимировичу, чьи советы оказали неоценимую помощь в достижении результатов, изложенных в данной работе.

**Литература**

1. Яблонский С .В., Введение в дискретную математику. – М.:Наука,1979.

2. Р. Лидл, Г. Нидеррайтер, Конечные поля. – М.:Мир, 1988.

3. М. И. Карагаполов, Ю.И. Мерзляков, Основы теории групп. – 3-е издание-М.:Наука, 1982.

4. М. А. Арбиб, Декомпозиция автоматов и расширение полугрупп // Алгебраическая теория автоматов, языков и полугрупп – М.“Статистика“, 1975, С. 46-64

5. Родин С. Б., Переходные системы с максимальной вариантностью относительно кодирования состояний // Интеллектуальные системы. Т.4, вып. 3-4. С. 335-352.

6. Родин С. Б., Линейно реализуемые переходные системы // Интеллектуальные системы. Т.14, вып. 1-4. С. 491-502.

# ГЕОМЕТРИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ РАСПОЗНАВАНИЯ АВТОМАТОВ

**Твердохлебов В.А. (г.Саратов)**
*TverdokhlebovVA@list.ru*

В работах 1995-1996г.г. [1-2] предложен, а в дальнейшем [3-5] разработан новый способ задания инициального дискретного детерминированного автомата A = (S, X, Y, $\delta$, $\lambda$, $s_0$) , где S, X и Y - множества состояний, входных и выходных сигналов, $\delta : S \times X \to S$ и $\lambda : S \times X \to Y$ - функции переходов и выходов, а $s_0$ - начальное состояние. Построены и проанализированы примеры эффективности предлагаемого способа задания автоматов, которые убедили не только автора (см.[6]), но и других специалистов в полезности нового способа задания автоматов. В связи с этим приводятся модели и методы, в которых представлен новый способ. Автомату A соответствуют эквивалентные автоматные отображения $\rho_s = \bigcup\limits_{p \in X^*} \{(p, \lambda(s,p))\}$ и $\rho'_s = \bigcup\limits_{p \in X^*} \{(p'x, \lambda(\delta(s,p'),x))\}$ . Предлагаемому способу задания автоматов соответствуют три вида структур - символьный график и два числовых графика с целочисленными и вещественными координатами точек. Символьный график автоматного отображения представляется в системе координат с осью абсцисс $(X^*, \omega_1)$ и осью ординат $(Y, \omega_2)$ , где $\omega_1$ и $\omega_2$ - линейные порядки на множествах X* и Y . В графике с положительными целочисленными координатами точек каждая точка (p, y) , где $p \in X^*$ и $y \in Y$ , преобразована в точку $(r_1(p), r_2(y))$, где $r_1(p)$ и $r_2(y)$ - номера p и y по линейным порядкам $\omega_1$ и $\omega_2$ . В связи с тем, что задание автоматного отображения числовыми графиками позволяет использовать в теории автоматов мощные идеализации непрерывной математики (актуальную бесконечность, бесконечно малую величину, непрерывность, предельный переход, суммирование бесконечных рядов и др.), разработан способ задания автомата, основывающийся на отображении вида $\phi_X : X^* \to R^+$ и $\phi_Y : Y \to \{\alpha_1, \alpha_2, ..., \alpha_l\}$, где $\alpha_i$ - полуинтервалы, составляющие единый полуинтервал на оси ординат , являющейся частью оси ординат первого квадранта прямоугольной декартовой системы координат на плоскости. В этом случае положение на оси абсцисс элемента $p \in X^*$ определяется точно, а элементу $y \in Y$ соответствует полуинтервал. При новом способе задания

инициального автомата моделью автоматного отображения является множество точек, представляющее автоматное отображение в целом и варианты сечений этого множества, выделяющие подмножества, соответствующие конкретным вариантам функционирования автомата. Числовая форма автоматного отображения позволяет располагать точки графика на геометрических кривых линиях, заданных аналитически. Это, в свою очередь, позволяет рассматривать свойства автоматного отображения как свойства геометрических кривых и использовать вычисления с применением уравнений и неравенств, связанных с геометрическими кривыми. Для линейных порядков $\omega_1$ и $\omega_2$ , соответствующих лексикографическому упорядочиванию, имеет место теорема.

**Теорема 1.** *Пусть* $p \in X^*$ *и* $p = x_{i_1}x_{i_2}...x_{i_k}$ *, где* $k \in N^+$ *. Тогда номер r(p) слова p по порядку* $\omega_1$ *определяется равенством:*

$$r(p) = \sum_{j=1}^{k} r(x_{i_j}) \cdot |X|^{j-1} - 1 \ .$$

Следующие теоремы устанавливают связи между произвольными геометрическими кривыми и результатами их интерпретации в представлении графиков автоматных отображений.

**Теорема 2.** *Пусть* $A = (S, X, Y, \delta, \lambda, s_0)$ *- инициальный дискретный детерминированный автомат с конечным или счетно-бесконечным множеством состояний S,* $\omega_1$ *-линейный порядок на* $X^*$ *и* $(\alpha_0, \alpha_l]$ *- полуинтервал на оси ординат, где* $l = |Y|$*. Тогда для любых*

*- взаимно-однозначного отображения "в"*$\phi : N^+ \to R$*, где для любых* $n, n' \in N^+$ *из* $n < n'$ *следует* $\phi(n) < \phi(n')$*;*

*- разбиения полуинтервала* $(\alpha_0, \alpha_l]$ *на l полуинтервалов* $(\alpha_0,\ \alpha_1]$ *,* $(\alpha_1,\ \alpha_2]$*, ... ,* $(\alpha_{l-1}, \alpha_l]$ *и взаимно-однозначного отображения*

$\nu : Y \to (\alpha_{i-1}, \alpha_i], 1 \le i \le l,$

*пара чисел* $(j, \beta)$*, где* $j \in Pr_2\phi$ *и* $\beta \in (\alpha_0, \alpha_l]$*, однозначно определяет пару* $(p, y_i)$*, для которой j - номер* $p \in X^*$ *по порядку* $\omega_1$ *и* $\beta \in (\alpha_{i-1}, \alpha_i]$*.*

**Теорема 3.** *Любые:*

*- геометрическая кривая* $y = f(x)$*;*

*- последовательность h точек* $(x_{i_1}, f(x_{i_1})), (x_{i_2}, f(x_{i_2})), ..., (x_{i_j}, f(x_{i_j})), ...,$ *где* $x_{i_1} < x_{i_2} < ...x_{i_j} < ...$*;*

*- число* $m \in N^+$ *и разбиение последовательности h на подпосле-*

*довательности из $m$ элементов каждая;*

*- полуинтервал $\Delta = (\alpha, \beta]$ на оси ординат, где $\min\limits_{x\in\Delta} f(x) < \alpha < \beta \leq \max\limits_{x\in\Delta} f(x)$;*

*- разбиение полуинтервала $\Delta$ на конечное множество полуинтервалов вида $(\alpha, \alpha_1]$ , $(\alpha_1, \alpha_2]$, ... , $(\alpha_{l-1}, \beta]$, где $l \in N^+$,*

*однозначно определяют геометрический образ законов функционирования дискретного детерминированного автомата с конечным или счетно-бесконечным множеством состояний, с $m$ входными и $l$ выходными сигналами.*

Методы распознавания инициальных автоматов, заданных числовыми графиками автоматных отображений, ориентированы на автоматы с большим, в общем случае счетно-бесконечным, числом состояний и построены на следующих процедурах.

Все варианты поведения инициального автомата представлены числовым графиком, точки которого предполагаются расположенными на геометрической кривой $y = f(x)$. Конкретное расположение точек на этой кривой определяется в соответствии с фактической информацией, как правило, частичной и дополняемой с использованием гипотез о свойствах процесса функционирования автомата.

Существует простой метод преобразования геометрического задания автоматного отображения в функции переходов и выходов и, если позволяют мощности множеств состояний, входных и выходных сигналов, в табличное задание автоматов.

Методу минимизации автомата по табличному заданию автомата соответствует простой и эффективный метод минимизации по геометрическому образу автомата. Можно утверждать, что известные методы анализа свойств автоматов имеют аналогичные методы действий с геометрическими образами автоматов. Основной метод распознавания автомата в классе автоматов, заданных числовыми графиками автоматных отображений, содержит следующие этапы (условие исключительности класса автоматов предполагается).

1 Этап. Для уравнений кривых $y_i = f_i(x), i \in I$ , на которых расположены точки автоматных отображений, строятся:

- неравенства $f_i(x) \neq f_j(x), i, j \in I$ и $i \neq j$ ;
- равенства $f_i(x) = f_j(x), i, j \in I$ и $i \neq j$ .

2 Этап. По равенствам определяются интервалы оси абсцисс, содержащие точки, определяющие входные последовательности, на ко-

торых автоматы не распознаются по наблюдаемому поведению.

3 Этап. По неравенствам определяются интервалы оси абсцисс, содержащие точки, определяющие входные последовательности, на которых автоматы распознаются по наблюдаемому поведению.

4 Этап. Анализируются результаты второго и третьего этапов и строится общая картина областей оси абсцисс (множеств входных последовательностей), в которых находятся входные последовательности - решения задачи распознавания автомата в рассматриваемом классе автоматов.

**Литература**

1. Твердохлебов В. А. Техническое диагностирование в геометрической интерпретации задач, моделей и методов // Материалы междунар. конф. Автоматизация проектирования дискретных систем. / Белорус. гос. ун-т, Ин-т техн. кибернетики АНБ. - Минск : Изд-во Белорус. гос. ун-та, 1995. - Т. 1 : Тезисы докладов. - с. 97.

2. Твердохлебов В. А. Распознавание автоматов на основе геометрической интерпретации // Проблемы теоретической кибернетики : тез. докл. XI Междунар. конф., 10-14 июня 1996 г. М. : Изд. РГГУ, 1996. - с. 85-93.

3. Твердохлебов В. А. Геометрические образы конечных детерминированных автоматов // Изв. Сарат. ун-та. Новая серия. Сер. Математика, Механика, Информатика. - 2005. - Т. 5, вып. 1. - с. 141-153.

4. Твердохлебов В. А. Методы интерполяции в техническом диагностировании. / Ж-л "Проблемы управления". М. N2 2007. с. 28-34.

5. Твердохлебов В. А. Геометрические образы законов функционирования автоматов.-Саратов: Изд-во "Научная книга 2008. 183с.

6. Безопасность критических инфраструктур: математические и инженерные методы анализа и обеспечения. Под.ред. В.С.Харченко. Харьков. Изд-во Национальный аэрокосмический университет им. Н.Е.Жуковского. ("ХАИ").2011г. 641с.

# СЛОЖНОСТЬ КОНСТРУИРОВАНИЯ ИЗОБРАЖЕНИЙ КЛЕТОЧНЫМИ АВТОМАТАМИ

**Титова Е.Е. (Московский Государственный Университет)**
*titovaee@yandex.ru*

В работе рассматривается задача конструирования изображений клеточным автоматом на прямоугольном экране. В каждую клетку прямоугольного экрана $n \times m$ помещено по одному экземпляру одного и того же автомата $\mathcal{A}$ (элементарного), к его входам присоединены выходы автоматов, стоящих в соседних клетках, выход автомата — его текущее состояние. Доопределим нулями крайние входы автоматов $n$-й строки и $m$-го столбца. Неопределенные входы автоматов первой строки и первого столбца будем называть свободными входами, а всю эту конструкцию — $(n, m)$-*экраном* $S =< \mathcal{A}, n, m >$. Также имеется внешний автомат $\mathcal{A}_e$ с $(n + m)$ выходами, который генерирует входные последовательности для свободных входов элементарных автоматов. Пара $G =< \mathcal{A}_e, S >$, состоящая из экрана и внешнего автомата называется *генератором*. Если каждый элементарный автомат экрана находится состоянии 0 или 1, то такую конфигурацию состояний экрана будем называть *черно-белой конфигурацией*. Черно-белую конфигурацию назовем *изображением*, если ее можно удерживать сколь угодно долго, подавая на свободные входы автоматов нулевые значения. *Кодом* $K$ изображения назовем матрицу $n \times m$, состоящую из нулей и единиц. Скажем, что изображение $\Im_K$ *соответствует данному коду* $K$, если положение нулей и единиц в изображении и в коде совпадают. Обозначим $\Im(n, m)$ — множество всех изображений размера $n \times m$. Экран $S =< \mathcal{A}, n, m >$ — *универсальный*, если для любого кода $K$ существует внешний автомат $\mathcal{A}_e^K$, т. ч. генератор $G =< \mathcal{A}_e^K, S >$ формирует изображение $\Im_K$, соответствующее коду $K$. $\mathcal{U}(n, m)$ — множество всех универсальных $(n, m)$-экранов. Через $\mathcal{G}(S, \Im)$ обозначим множество генераторов $< \mathcal{A}_e, S >$, формирующих изображение $\Im$. Если $S =< \mathcal{A}, n, m >$ — экран, то $Q(S)$ — число состояний элементарного автомата $\mathcal{A}$, $Q(n, m) = \min_{S \in \mathcal{U}(n,m)} Q(S)$.

Пусть $G =< \mathcal{A}_e, S >$ — некоторый генератор. Автомат $A_e$ получает на вход некоторую последовательность, содержащую информацию о коде изображения, которое должен построить генератор. Эту последовательность будем генерировать по коду $K$ с помощью следу-

ющих устройств: перестановка $\pi$, разреживатель с коэффициентом $d \leq 1$, разреживатель с коэффициентом $d \geq 1$, задержка $G_a^k$.

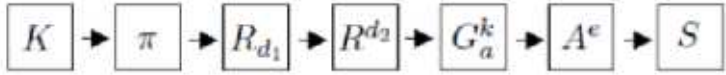



Рис. 1. Схема построения входных последовательностей для свободных входов экрана.

*Перестановкой* $\pi$ будем называть отображение матрицы $K$ (кода изображения) в некоторый вектор, элементами которого для разных алгоритмов могут быть нули и единицы, либо пары координат некоторых элементов кода изображения (например тех, которые равны 1), либо какая-то другая информация, взятая из $K$.

Пусть имеется некоторое множество $V$. *Разреживателем* $R_d$ с коэффициентом $d, d = 1/s, s \in \mathbf{N}$, будем называть отображение из множества конечномерных векторов над $V$ в множество конечномерных векторов над $V^{1/d}$, такое что если $v = (v_1, \ldots, v_{k \cdot s}) \in V^k$, то $R_d(v) = ((v_1, \ldots, v_s), \ldots, (v_{(k-1)s+1}, \ldots, v_{ks}))$, т.е. элементами нового вектора являются векторы, состоящие из $s$ элементов вектора $v$. *Разреживателем* $R^d$ с коэффициентом $d, d \in \mathbf{N}$ будем называть отображение из множества конечномерных векторов над $V$ в множество конечномерных векторов над $V \cup \{0\}$ (возможно $0 \in V$), такое что если $v = (v_1, \ldots, v_k) \in V^k$, то

$$R^d(v) = (v_1, 0, \ldots, 0, v_2, 0 \ldots, 0, \ldots, v_{k-1}, 0, \ldots, 0, v_k),$$

где между всеми элементами вектора $v$ вставлено по $(d-1)$-му новому элементу, каждый из которых для вектора $R^d(v)$ равен 0.

*Задержкой* $G_a^k, k \in \mathbf{N} \cup \{0\}$ для конечной последовательности элементов из множества $V$ (возможно $a \in V$), будем называть устройство, которое, получая на вход эту последовательность, в первые $k$ тактов выдает элемент $a$, а затем по порядку все полученные на вход элементы.

Итак, на вход перестановки $\pi$ поступает код $K$ некоторого изображения. $\pi$ формирует из него некоторый информационный вектор, который поступает на вход разреживателя $R_{d_1}$. $R_{d_1}$ объединяет элементы входного вектора в группы по $d_1$ элементов и подает на вход разреживателю $R^{d_2}$, который между этими информационными эле-

ментами вектора вставляет нулевые элементы. Полученный вектор поступает на вход задержки $G_a^k$, которая в начало вектора добавляет $k$ элементов $a$. Полученная таким образом последовательность поступает на вход внешнего автомата $A_e$, который генерирует последовательности для свободных входов экрана. Описанная схема изображена на рисунке 1.

*Алгоритмом построения изображений на заданном универсальном экране* будем называть множество последовательностей входных элементов для свободных входов экрана, при подаче которых на экране формируются наперед заданные соответствующие им изображения. Скажем, что генератор $G =< \mathcal{A}_e, S >$ соответствует алгоритму $\mathcal{A}$ построения изображений на универсальном экране $S$, если для любого $K$ автомат $\mathcal{A}_e$ (получая на вход построенную по коду $K$ с помощью описанной выше схемы последовательность) генерирует последовательность, соответствующую изображению $\Im_K$ в алгоритме $\mathcal{A}$. Пусть $\mathcal{A}$ — алгоритм построения изображений на универсальном экране $S(n, m)$. Множество всех генераторов, соответствующих алгоритму $\mathcal{A}$ обозначим $\mathcal{G}(\mathcal{A})$. Если $G =< \mathcal{A}_e, S(n, m) >\in \mathcal{G}(\mathcal{A})$, то обозначим $Q_e(G)$ — число состояний внешнего автомата $\mathcal{A}_e$, $Q(G) = Q(S) \cdot Q_e(G)$ — сложность генератора $G$. Сложностью алгоритма $\mathcal{A}$ назовем $Q(\mathcal{A}) = \min_{G\in\mathcal{G}(\mathcal{A})} Q(G)$.

В [3] и [4] приведены алгоритмы построения изображений на универсальных экранах с $Q(S) = 3$ (*Алгоритм 3*), $Q(S) = 4$ (*Алгоритм 4*), $Q(S) = 5$ (*Алгоритм 5*), $Q(S) = 2n + 2$ (*Алгоритм* $A_{min}$), также приведен *Алгоритм 7* построения изображения на экране с одним свободным входом. Будем обозначать эти алгоритмы $\mathcal{A}3$ , $\mathcal{A}4$, $\mathcal{A}5$, $\mathcal{A}_{min}$ и $\mathcal{A}7$ соответственно.

Приведем здесь *Алгоритм 3*. Без ограничения общности будем считать, что $m \geq n$. Опишем элементарный автомат. Множество состояний — $E_q = \{0, 1, 2\}$. Функцию переходов состояний зададим следующим образом: $\varphi(q, 2, r, u, d) = 2$ для любых $q, r, u, d \in \{0, 1\}$; $\varphi(2, l, r, 2, d) = 2$ для любых $l, r, d \in \{0, 2\}$; $\varphi(2, l, r, u, d) = u$ для любых $u \in \{0, 1\}$, $d \in \{0, 1, 2\}$, $l, r\{0, 2\}$; $\varphi(q, l, r, 2, 2) = 2$ для любых $l, q, r \in \{0, 1\}$ $d \in \{0, 1\}$; $\varphi(q, l, r, u, d) = q$ в остальных случаях.

Опишем выходы внешнего автомата, соответствующие изображению с кодом $K$. Первый выходной вектор (длины $n$) в первый такт будет равен $(2, 2, \ldots, 2)_n$, во второй и все последующие такты первый выходной вектор будет нулевым, т.е. $(0, 0, \ldots, 0)_n$. Вектор

$(b_1, \ldots, b_m)$, подаваемый на верхнюю границу экрана будем строить по следующим правилам: в первый такт это нулевой вектор, т.е. $(0, 0, \ldots, 0)_m$; далее при $2 \le i \le m+1$ в $i$-й такт в первых $(i-1)$ битах стоят 2, в остальных битах вектора стоят нули, т.е. $(2, ..., 2, 0, ..., 0)_m$; при $m + 2 \le i \le m + 2n$: если $i = (m + 2) + 2s$, $0 \le s \le n - 1$, то в $i$-й такт выходной вектор равен $(m - s)$-й строке кода $K$; если $i = (m+2)+2s+1$, $0 \le s \le n-1$, то в $i$-й такт в каждом бите выходного вектора стоит 2, т.е. он равен $(2, 2, \ldots, 2)_m$; при $i \ge m + 2n + 1$ выходной вектор нулевой, т.е. $(0, 0, \ldots, 0)_m$.

При подаче выходов описанного внешнего автомата на свободные входы экрана состояние 2 распространяется по горизонтали слева направо, пока не заполнит весь экран. Далее на верхнюю границу с задержкой в один такт подаются друг за другом строки кода изображения начиная с нижней и заканчивая верхней строкой. Такая строка кода продвигается по экрану вниз до тех пор, пока и сверху и снизу от нее находятся строки из двоек. Если соседняя снизу строка состоит из нулей и единиц, то строка кода останавливается и дальше не двигается. Таким образом на экране снизу вверх по строкам восстанавливается изображение, соответствующее заданному коду. Оно появляется на экране в следующий такт после подачи на свободные входы экрана последнего ненулевого вектора.

Имеют место следующие оценки сложности алгоритмов построения изображений на универсальных экранах.

**Теорема 1** *Если* $< \mathcal{A}_e, S(n,m) > \in \mathcal{G}(\mathcal{A}3)$, $n, m \in \boldsymbol{N}, n \le m$, *то* $Q_e(G) \le m + 2, Q(\mathcal{A}3) \le 3m + 6$.
*Если* $< \mathcal{A}_e, S(n,m) > = G \in \mathcal{G}(\mathcal{A}4)$, $n, m \in \boldsymbol{N}, n \le m$, *то* $Q_e(G) \le 2n, Q(\mathcal{A}4) \le 8n$.
*Если* $< \mathcal{A}_e, S(n,m) > = G \in \mathcal{G}(\mathcal{A}5)$, $n, m \in \boldsymbol{N}, n \le m$, *то* $Q_e(G) \le 3, Q(\mathcal{A}5) \le 15$.
*Если* $< \mathcal{A}_e, S(n,m) > = G \in \mathcal{G}(\mathcal{A}_{min})$, $n, m \in \boldsymbol{N}, n \le m$, *то* $Q_e(G) \le n, Q(\mathcal{A}_{min}) \le 2n^2 + 2n$.
Если $< \mathcal{A}_e, S(n,m) > = G \in \mathcal{G}(\mathcal{A}7)$, $n, m \in \mathbf{N}, n \le m$, то $Q_e(G) \le 6, Q(\mathcal{A}7) \le 48$.

Автор выражает глубокую признательность научному руководителю д.ф.-м.н., профессору Э.Э. Гасанову за постановку задачи и научное руководство.

**Литература**

1. В. Б. Кудрявцев, А. С. Подколзин, А. А., Болотов Основы теории однородных структур // Москва, "Наука 1990.

2. В. Б. Кудрявцев, С. В. Алешин, А. С. Подколзин, Введение в теорию автоматов // Москва, "Наука 1985.

3. Е. Е. Титова, Конструирование изображений клеточными автоматами // Интеллектуальные системы, том 12, вып. 1-4, стр. 105-121, Москва, 2008.

4. Е. Е. Титова, Линейное по времени конструирование изображений клеточными автоматами // Интеллектуальные системы, том 15, вып. 1-4, Москва, 2011.

# ДИСКРЕТНЫЕ ДИНАМИЧЕСКИЕ СИСТЕМЫ, ОПРЕДЕЛЯЕМЫЕ ГЕОМЕТРИЧЕСКИМИ ОБРАЗАМИ АВТОМАТОВ

**Тяпаев Л.Б., Василенко Д.В. (Саратовский государственный университет)**

*TiapaevLB@info.sgu.ru*

Объектом исследования является динамическая система, определяемая геометрическими образами автоматов. Фазовое пространство системы определяется ортогональными и аффинными преобразованиями геометрических образов. Изучаются произведения динамических систем заданного типа и их характеристики.

*Динамической системой* называется тройка объектов $D = (S,\ f,\ G)$, где $S$ – пространство состояний, $|S| < \infty$, $f : S \longrightarrow S$ – функция эволюции, $G$– граф фазового пространства динамической системы.

В дальнейшем будем использовать термин *точка динамической системы* в качестве синонима для термина состояние динамической системы.

*Аттракторами* динамической системы $D = (S,\ f,\ G)$ называют циклы графа $G$. *Точкой ветвления* динамической системы $D$ называется вершина графа $G$, в которую входит более чем одна ветвь. *Стволом* дерева называется ветвь, содержащая точки ветвления.

*Произведением* динамических систем $D_1 = (S, f, G)$ и $D_2 = (T, g, G')$ будем называть динамическую систему $D_1 \circ D_2 = (S \times T, f \times g,\ G \circ G')$, где $S \times T$ - декартово произведение множеств $S$ и $T$; $f \times g : S \times T \longrightarrow S \times T$ - прямое произведение отображений $f : S \longrightarrow S$ и $g : T \longrightarrow T$, $f \times g : (s,t) \mapsto (f(s), g(t))$; $G \circ G'$ - граф определяемый следующим образом:

1. $V(G \circ G') = VG \times VG'$;

2. Вершины $(u, u')$ и $(v, v')$ графа $G \circ G'$ смежны тогда и только тогда, когда одновременно вершины $u$ и $v$ смежны в графе $G$, и вершины $u'$ и $v'$ смежны в графе $G'$.

Будем рассматривать динамические системы, определяемые геометрическими образами автоматов [3]. Пространство состояний динамической системы – конечное множество геометрических образов конечных автоматов. Геометрические образы автоматов суть множества точек плоскости с рациональными координатами, прообразами

которых являются множества входных слов автомата и его реакций [1,2]. Фазовое пространство динамической системы формируется посредством ортогональных и аффинных преобразований пространства состояний [3].

Дадим необходимые определения.

*Конечный автомат* это пятёрка $A = (S,\ X,\ Y, \delta\ ,\ \lambda)$, где

$S = \{s_0,\ s_1,\ \ldots,\ s_{N-1}\}$ – множество состояний автомата,

$X = \{x_1,\ x_2,\ \ldots,\ x_L\}$ – множество входных символов (входной алфавит), $Y = \{y_1,\ y_2,\ \ldots,\ y_M\}$ - множество выходных символов (выходной алфавит), $\delta : S \times X \ \rightarrow S$ – функция переходов автомата, $\lambda : S \times X \ \rightarrow Y$ – функция выходов автомата. Расширим функции $\delta$ и $\lambda$ на словах из множеств $X^*$ и $Y^*$ соответственно, и в дальнейшем будем использовать те же обозначения для этих функций. Здесь $X^*$ и $Y^*$ - множества слов конечной длины над алфавитами $X$ и $Y$ соответственно.

Пусть $s_0$ – начальное состояние автомата $A$. *Инициальным автоматом* называется пара $(A,\ s_0)$. Автомат $A$ называется *автономным*, если $|X| = 1$.

*Поведение автомата* $A$ определяется множеством

$$\Lambda = \{(p, q)\,|p \in X^* \exists s \in S\ \&q = \lambda\,(s, p)\}\,.$$

Множество $\Lambda_A(s_0) = \{(p, q)\,|p \in X^*, q \in Y^* \quad \lambda\,(s_0, p) = q\}$ определяет поведение автомата A из состояния $s_0$.

*Геометрическое пространство* $\Gamma$ для автомата $(A, s_0)$ определяется следующим образом [1]:

1. Сопоставим элементам множества $X$ натуральные числа от 1 до $L$, т.е. осуществим взаимно однозначное отображение $f : X \longrightarrow \{1, 2, \ldots, L\}$.

2. Определим координатную ось абсцисс $\tilde{X}$ для пространства $\Gamma$ как отрезок числовой оси $[0, L + 1]$.

3. Каждому слову $p = x_{i_1} x_{i_2} \ldots x_{i_k}$ сопоставим вектор $\omega = (f\,(x_{i_1})\,, f\,(x_{i_2})\,, \ldots, f\,(x_{i_k}))$, т.е. осуществим взаимно однозначное соответствие $g : X^* \longrightarrow V_N$, где $V_N$ - пространство конечномерных векторов, элементами которых являются натуральные числа.

4. Каждому такому вектору $\omega = (\omega_1,\ \omega_2, \ldots,\ \omega_k)$ взаимно однозначно сопоставим точку $\tilde{x} \in \mathbb{Q}$ на оси абсцисс:

$$\tilde{x} = \frac{\omega_1}{(L+1)^0} + \frac{\omega_2}{(L+1)^1} + \frac{\omega_3}{(L+1)^2} + \cdots + \frac{\omega_k}{(L+1)^{k-1}}.$$

Аналогично определяется нумерация элементов множества $Y$, ось ординат $\tilde{Y}$ пространства $\Gamma$ и отображение $h : Y^* \longrightarrow V_N$.

Каждой паре $(p, q) \in \Lambda_A(s_0)$ в пространстве $\Gamma$ сопоставляется точка с координатами $(\tilde{x}, \tilde{y})$, где $\tilde{x} = \sum_{i=1}^{|p|} \frac{c_i}{(L+1)^{i-1}}$, $\big(c_1, c_2, \ldots, c_{|p|}\big) = g(p)$,
$\tilde{y} = \sum_{i=1}^{|q|} \frac{b_i}{(M+1)^{i-1}}$, $\big(b_1, b_2, \ldots, b_{|q|}\big) = h(q)$.

Под *геометрическим образом* $\Omega_{\mathrm{A}}(\mathrm{s}_0)$ автомата $(A, s_0)$ понимается множество таких пар $(\tilde{x}, \tilde{y})$. *Кривой $f$, задающей поведение автомата* $(A, s)$, называется любая непрерывная кривая, такая, что любая точка $(\tilde{x}, \tilde{y}) \in \Omega_A(s)$ принадлежит кривой $f$. Кривая $f$ называется *функциональной кривой*, если $f$ есть график некоторой непрерывной функции. Понятие функции, определяющей функциональную кривую, отождествляется с самой кривой. Будем обозначать класс $K(N, L, M)$ автоматов, у которых $|S| = N$, $|X| = L$, и $|Y| = M$. Класс автономных автоматов обозначим $K(N, M)$. Аналитическое задание геометрического образа автономного автомата характеризует следующая теорема.

**Теорема [2]**. *Пусть $A \in K(N, M)$. Тогда поведение автомата $(A, s)$ в пространстве $\Gamma$ можно определить функциональной кривой $f$, которая может быть задана следующим уравнением:*

$$f(\tilde{x}) = \sum_{j=1}^{M} \left( j \cdot \sum_{i=1}^{l_j} (M+1)^{\Delta_i^{(j)} - \log_2 \frac{1}{2-\tilde{x}}} \right), \text{ где } 0 \le l_j \le N,$$
$$\Delta_i^{(j)} = (N-1) - r_i^{(j)},\ r_i^{(j)} \in \{0, 1, ..., N-1\}.$$

Две кривые называются *аффинно-эквивалентными*, если они могут быть получены одна из другой с помощью аффинного преобразования. Совокупность всех кривых, аффинно-эквивалентных какой-нибудь определенной кривой $f$, называется *аффинным классом* кривой $f$.

Зафиксируем некоторый класс автономных автоматов $K(N, M)$ и рассмотрим множество $\Omega$ всех различных геометрических обра-

зов из данного класса. Будем рассматривать аффинные преобразования, которые преобразуют некоторый образ $\Omega_i \in \Omega$ в другой образ $\Omega_j \in \Omega$. При рассмотрении преобразований геометрических образов из одного класса $K(N, M)$ имеет смысл рассматривать только следующее преобразование: параллельный перенос вдоль оси ординат и растяжение и сжатие относительно оси абсцисс. Данное преобразование имеет вид: $\tilde{x}' = \tilde{x}, \tilde{y}' = a\tilde{y} + b, \quad a, b \in \mathbb{Q}$. Тогда образ $\Omega_i \in \Omega$ переводится в образ $\Omega_j \in \Omega$ описанным преобразованием с коэффициентами $(a, b)$, если $(\forall(\tilde{x}, \tilde{y}) \in \Omega_i\ )\,((\tilde{x}, a\tilde{y} + b) \in \Omega_j)$. Будем говорить, что образы $\Omega_i, \Omega_j$ *совместимы* выбранным видом аффинного преобразования. Бинарное отношение $\rho \subseteq \Omega^2$, образованное парами совместимых образов

$$\rho = \{(\Omega_i, \Omega_j) \in \Omega | \exists a, b \in \mathbb{Q}\,(\forall\,(\tilde{x}, \tilde{y}) \in \Omega_i)\,((\tilde{x}, a\tilde{y} + b) \in \Omega_j)\}$$

является отношением эквивалентности на множестве $\Omega$ и задает разбиение этого множества на классы эквивалентности. Определен вид коэффициентов $(a, b)$ афинных преобразований геометрических образов $\Omega$, и установлено, что множество коэффициентов афинных преобразований для классов $K(N, M)$, $K(2, M)$ и $K(N, L, M)$ совпадают [4].

Рассмотрим максимальный класс эквивалентности $K$ отношения совместимости $\rho \subseteq \Omega^2$ геометрических образов автоматов из класса $K(N, M)$. Выберем произвольным образом периодическую последовательность $u(v)$ элементов данного класса, где $u$ и $v$ – конечные последовательности различных элементов. Каждая такая последовательность $u(v)$ порождает последовательность $F$ преобразований геометрических образов автоматов из класса $K$. Построим динамическую систему $D$, состояниями $S$ которой будут элементы последовательности $u(v)$, а эволюция состояний будет определяться последовательностью $F$ преобразований.

Рассмотрим свойства операции произведения динамических систем. Граф динамической системы состоит из циклов-аттракторов и притягиваемых деревьев.

Обозначим через $O_m + P_n$ динамическую систему, граф которой представляет собой цикл-аттрактор длины $m$ и притягиваемую цепь длины $n$, а через $P^m_{n,q}$ притягиваемое дерево, содержащее $m$ цепей

длины $n$, где $q$ – номер первой точки ветвления ствола данного дерева. Нумерация вершин ствола предполагается от листа к корню.

*Расстоянием между корнями* притягиваемых деревьев будем называть минимальную длину пути между ними.

**Теорема.** *Пусть динамическая система $D$ имеет структуру $O_m + P_n$. Тогда каждая из компонент связности графа системы $D \circ D$ представляет собой структуру вида $P^m_{n,q_1} + O^i_m + P^m_{n,q_2}$, где $q_1$ и $q_2$ – номера первых точек ветвления для первого и второго дерева соответственно, $i$ – расстояние между корнями деревьев, и обладает следующими характеристиками:*

*1. Количество компонент связности равно $m$. Длина цикла в каждой компоненте связности равна $m$. В каждой из компонент связности присутствует два притягиваемых дерева, каждое из которых имеет ствол. В каждой компоненте связности удалённость всех листов деревьев от аттрактора равна $n$.*

*2. Точками ветвления являются вершины графа, которые определяются точкой вхождения притягиваемой цепи в аттрактор системы $D$ и точкой, находящейся вне аттрактора системы $D$, и располагаются на стволах с периодичностью, равной $m$. Первые точки ветвления для каждой компоненты связности вычисляются по формулам: $q_1 = n - m + i$, $q_2 = n - i$, где $q_1$ и $q_2$ – номера элементов ствола, являющихся первым точками ветвления для первого и второго дерева. Если $q_i < 0$, то искомая точка ветвления не существует.*

*3. В одной компоненте связности расстояние между корнями притягиваемых деревьев равно 0 и в одной компоненте связности расстояние между корнями притягиваемых деревьев равно $m/2$ (для чётных $m$). Во всех остальных компонентах связности расстояние между корнями меняется от 1 до $m/2-1$ для чётных $m$ и до $(m-1)/2$ для нечётных $m$ с шагом 1, причём каждая из таких компонент будет встречаться в полученной динамической системе дважды.*

*4. Если расположить ствол деревьев перпендикулярно входящим в них цепям, они образуют прямоугольные равнобедренные треугольники. При таком способе отображения компоненты связности с расстоянием между корнями деревьев, равным 0 и $m/2$ (для чётных $m$) обладают свойством симметрии относительно вертикали.*

**Литература**

1. Тяпаев Л. Б. Геометрическая модель поведения автоматов и их неотличимость // Математика, Механика, Математическая кибернетика. Сб. науч. тр. – Саратов: Изд-во Сарат. ун-та, 1999. – С. 139–143.

2. Тяпаев Л. Б. Решение некоторых задач для конечных автоматов на основе анализа их поведения // Изв. Сарат. ун-та. Сер. Математика. Механика. Информатика, 2006. – Т.6., Вып.1/2 – С. 121-133.

3. Тяпаев Л. Б. Геометрические образы автоматов и динамические системы // Дискретная математика и ее приложения. Материалы X Межд. семинара. Под ред. О.М. Касим-Заде. – М.: Изд-во механико-математического факультета МГУ, 2010. – С. 510-513.

4. Матов Д. О. Аффинные преобразования геометрических образов конечных автоматов // Проблемы теоретической кибернетики: Материалы XVI Межд. конф. Под ред. Ю.И. Журавлева– Нижний Новгород: Изд-во Нижегородского госуниверситета, 2011. – С. 303-306.

# О ПОЛНОТЕ В КЛАССЕ КОНЕЧНЫХ АВТОМАТОВ, ВЫЧИСЛЯЮЩИХ НЕКОТОРЫЕ АФИННЫЕ ФУНКЦИИ

**Часовских А.А. (Москва)**
*chasovskikh@mail.ru*

Все необходимые определения можно найти в работах [1], [2]. Известно, что в классе $P_{\text{О.Д.}}$ всех ограниченно-детерминированных функций, рассматриваемых вместе с операциями композиции, проблема проверки полноты конечных множеств алгоритмически неразрешима. Все-же в $P_{\text{О.Д.}}$ содержатся нетривиальные подклассы, для которых указанная проблема алгоритмически разрешима. К ним относится, например, подкласс линейно-автоматных функций. Здесь приведен пример другого содержательного подкласса $P_{\text{О.Д.}}$ с разрешимой задачей о полноте.

Бесконечной последовательности нулей и единиц $\alpha$, $\alpha = \alpha(0)$, $\alpha(1)$, ..., $\alpha(t)$, ..., $\alpha(t) \in E_2$, $t = 0, 1, ...$ сопоставим формальный ряд

$$\bar{\alpha} = \sum_{t=0}^{\infty} \alpha(t) 2^t.$$

Положим

$$R = \{ \quad \bar{\alpha} \quad | \quad \alpha \in E_2^{\infty} \quad \},$$

$$PR = \{ \quad \bar{\alpha} \quad | \quad \alpha \in E_2^{\infty},$$
$$\alpha - \text{периодическое (с предпериодом) сверхслово} \quad \}.$$

На множестве $R$ введем операции сложения и умножения,при этом $\bar{\alpha}_1$ и $\bar{\alpha}_2$ суммируются или перемножаются как числа в двоичной записи с младшими разрядами $\alpha_1(0)$ и $\alpha_2(0)$.

Множество $PR$ совпадает с кольцом рациональных чисел, которые, будучи представленными в несократимом виде, имеют нечетный знаменатель.

Нетрудно видеть, что любой автомат с входным алфавитом $E_2^n$ и выходным алфавитом $E_2$ задает отображение из $PR^n$ в $PR$. Например, для задержки $\xi_1(x)$ с начальным состоянием 1 имеем: $\xi(\bar{\alpha}) = 1 + 2\bar{\alpha}$ для любого $\bar{\alpha} \in R$.

Конечный автомат с входным алфавитом $E_2^2$, задаваемый следующей системой канонических уравнений

$$\begin{cases} q(0) = 0, \\ q(t+1) = x_1(t) \wedge x_2(t) \vee q(t) \wedge x_1(t) \vee q(t) \wedge x_2(t), \\ y(t) = x_1(t) \oplus x_2(t) \oplus q(t) \end{cases}$$

осуществляет отображение $F_+^{(2)}$ из $R^2$ в $R$ по следующему правилу:

$$F_+^{(2)}(\bar{\alpha}_1, \bar{\alpha}_2) = \bar{\alpha}_1 + \bar{\alpha}_2.$$

Через $L$ обозначим замыкание множества $\left\{\xi_1(x), F_+^{(2)}(x_1, x_2)\right\}$ по операциям композиции.

Для любого $n$, $\quad n \in \{0, 1, \dots\}$, и для любой $F(x_1, x_2, \dots, x_n)$ найдутся $r_i$, $\quad$, $r_i \in PR$, $\quad$, $i = 0, 1, \dots, n$, такие, что для любых $\bar{\alpha}_i$, $\bar{\alpha}_i \in R$, $i = 1, 2, \dots, n$, выполнено:

$$F(\bar{\alpha}_1, \bar{\alpha}_2, \dots, \bar{\alpha}_n) = \sum_{i=1}^{n} r_i \bar{\alpha}_i + r_0. \tag{1}$$

Пусть выполнено (1). Положим

$$U(F) = \{ \quad r_i \quad | \quad i = 1, 2, \dots, n \quad \}.$$

Для множества $M$, $\quad M \subseteq L$, положим:

$$U(M) = \cup_{F \in M} U(F).$$

Переменная $x_i$ функции $F(x_1, x_2, \dots, x_n)$, удовлетворяющей равенству (1), называется существенной, если $r_i \neq 0$. Переменная $x_i$ называется непосредственной, если $r_i$, будучи представленным в несократимом виде, имеет нечетный знаменатель.

Операция обратной связи применима к переменной $x_i$ функции $F(x_1, x_2, \dots, x_n)$ в точности тогда, когда $x_i$ не является непосредственной переменной.

Далее, рассматривая дроби $p/q$ из $PR$, считаем, что $(p, q) = 1$. Положим:

$$H^1 = \{ \quad 1 + 2 \cdot p/q \quad | \quad p/q \in PR \quad \}.$$

Рассмотрим следующие подмножества в $L$.

$$L_c^1 = \{ \quad F \quad | \quad F \quad \text{имеет ровно одну существенную переменную} \quad \},$$

$$L_n^1 = \{ \quad F \quad | \quad F \quad \text{имеет ровно одну непосредственную переменную} \quad \},$$

$$T_a = \{ \quad F \quad | \quad \text{для любых} \quad \alpha_i, \quad \alpha_i \in E_2^\infty, \quad \alpha_i(0) = a, \quad i = 1, 2, \ldots, n, \quad \text{выполнено} \quad F(\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_n)(0) = a \quad \}, \quad a = 0, 1,$$

$$V_1 = \{ \quad F \quad | \quad F \quad \text{имеет не более одной существенной переменной} \quad \},$$

$$V_n = \{ \quad F \quad | \quad F \quad \text{имеет нечетное число непосредственных переменных} \quad \},$$

$$J = \left\{ \quad F \quad | \quad \sum_{r \in U(F)} |r| \leq 1 \quad \right\},$$

Пусть $p_1$,$p_2$,... - последовательность всех простых чисел, причем $p_1 < p_2 < \ldots$. Тогда $p_1 = 2$. Положим:

$$R_c(p_i) = \{ \quad F \quad | \quad F \in L \setminus L_c^1, \quad \text{для любого} \quad p/q \quad \text{из} \quad U(F) \quad \text{выполнено:} \quad (p, p_i) = p_i \quad \} \quad \cup \quad \{ \quad F \quad | \quad F \in L_c^1, \quad \text{для любого} \quad p/q \quad \text{из} \quad U(F) \setminus \{0\} \quad \text{выполнено:} \quad (q, p_i) = 1 \quad \}, \quad i = 2, 3, \ldots,$$

$$R_n(p_i) = \{ \quad F \quad | \quad F \in L \setminus L_n^1, \quad \text{для любого} \quad p/q \quad \text{из} \quad U(F) \quad \text{выполнено:} \quad (p, p_i) = p_i \quad \} \quad \cup \quad \{ \quad F \quad | \quad F \in L_n^1, \quad \text{для любого} \quad p/q \quad \text{из} \quad U(F) \setminus H^1 \quad \text{выполнено:} \quad (p, p_i) = p_i, \quad \text{а для любого} \quad p/q \quad \text{из} \quad U(F) \cap H^1 \quad \text{имеет место} \quad (q, p_i) = 1 \quad \}, \quad i = 2, 3, \ldots.$$

Через $A$ обозначим следующее множество:

$$\{ \quad T_0, T_1, V_1, V_n, J, R_c(p_i), R_n(p_i) \quad | \quad i = 2, 3, \ldots \quad \}.$$

Имеют место:

**Теорема 1.** *Множество $A$ является приведенной критериальной системой, состоящей из предполных в $L$ классов.*

**Теорема 2.** *Задачи проверки а-полноты и полноты конечных систем из $L$ алгоритмически разрешимы.*

Автор выражает благодарность академику В. Б. Кудрявцеву за постоянную поддержку в работе.

**Литература**

1. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Элементы теории автоматов. — М.: мзд-во МГУ, 1978.

2. Часовских А. А. О полноте в классе линейных автоматов // Математические вопросы кибернетики. — М.: Наука, 1991. —Вып. 3. —С. 140-166.

# О ХАРАКТЕРИЗАЦИИ СОСТОЯНИЙ АВТОМАТНОЙ МОДЕЛИ ЛЁГКИХ В ЧИСТОЙ СРЕДЕ

**Чернова Ю.Г. (Московский Государственный Университет им. М.В. Ломоносова)**
*yulyaha@list.ru*

В предлагаемой работе продолжается изучение функционирования легких человека, начатое в работах [2], [3] и [4]. В качестве главной функции легких рассматривается свойство их самоочищения как от внутреннего секрета, так и от поступающего извне вещества в легкие. В работе [2] установлено, что процесс самоочищения легких может быть промоделирован с помощью автоматов. Здесь мы изучаем свойства таких автоматов.

Легкие образуют древовидную структуру бронхов, в которых имеются реснички, играющие роль эскалаторного механизма вывода как внутреннего секрета, так и привнесенного извне вещества во внешнюю среду. Бронхи имеют разные пропускные способности и разную эффективность ресничек. Чем выше от самых мелких бронхов, тем мощнее механизм передачи вещества изнутри вовне.

Предполагается, что в момент $t$ распределено некоторое количества вещества по ресничкам легких. Тогда в момент $t+1$ по определенным правилам происходит перемещение вещества в легких с помощью ресничек по направлению к трахее. Этот процесс продолжается до полного освобождения легких от этого вещества.

Возникает задача построения модели легочного механизма самоочищения как в условиях чистой среды, так и в условиях возможного запыления легких из среды в процессе дыхания, а также задача изучения свойств этой модели.

Ранее автором была построена такая модель процесса самоочищения запыленных легких в предположении чистой среды [3], в которой оно функционирует. Как отмечено выше, автором было показано, что такая модель является автоматной, и потому разные свойства процесса самоочищения могут быть исследованы с помощью изучения соответствующих автоматов.

Основной характеристикой автоматов, как известно [1], является его диаграмма Мура, построение и изучение свойств которой для модели легких означало бы установление свойств процесса их самоочищения.

В предлагаемой работе проводится изучение этой диаграммы Мура. В качестве характерных состояний диаграммы выбраны те состояния, которые имеют наибольшее число предшественников. Такие состояния называются состояниями конденсации. Содержательно, это те состояния, для которых предыстория наиболее неопределена.

Основными результатами предлагаемой работы являются критериальное описание таких состояний конденсации, нахождение количества этих состояний, а также получение оценок для количества их предшественников.

Представим легкие полным дихотомическим ориентированным к корню деревом, которое будем называть *I-деревом* и обозначать $D^{-1}$, со следующими параметрами.

Пусть $\mathbb{N}$ - множество натуральных чисел и $l$, $l \in \mathbb{N}$, считаем глубиной этого I-дерева. Полагаем, что ребро I-дерева $D^{-1}$, инцидентное корню, имеет глубину 1.

Каждое ребро из $D^{-1}$ разделено на $n$, $n \in \mathbb{N}$, равных частей, называемых *ресничками*, и занумерованных числами $i$, $i = 1, 2, \ldots, n$, возрастающими в направлении, обратном ориентации ребра.

Припишем каждому ребру глубины $j$, $j = 1, 2, \ldots, l$, два числа $2^{l-j}b$ и $2^{l-j}r$, где $b, r \in \mathbb{N}$ и $r \leq b$, называемых *максимальной нагрузкой* и *мерой переброса* ресничек ребер глубины $j$ соответственно.

Такое I-дерево $D^{-1}$ с описанными выше параметрами $b, r, n$ и $l$ обозначим $D^{-1}(b, r, n, l)$.

Свяжем с ним некоторый процесс, который назовем *процессом дыхания*. Он обусловлен рядом допущений.

Считаем, что в $D^{-1}(b, r, n, l)$ заданы распределения значений нагрузок по всем ресничкам, учитывая, что нагрузка может быть нулевой. Пусть $V'$ — суммарная нагрузка по всем ресничкам, а $V$ — максимально возможная суммарная нагрузка по всем ресничкам. $V$ назовем *объемом I-дерева (легких)*, а $V'$ — *исходным объемом загруженности I-дерева*.

I-дерево $D^{-1}(b, r, n, l)$ с исходным объемом загруженности $V'$ обозначим $D^{-1}(b, r, n, l; V')$.

Каждая ресничка осуществляет прием вещества извне и переброс своей нагрузки на следующую ресничку с меньшим номером внутри ребра.

*Прием ресничкой вещества*, имеющего массу $d$, $d \in \mathbb{N}_0$ и $d \leqslant$

$V - V'$, из внешней среды внутри ребра осуществляется по следующему правилу (для этого правила ориентация считается обратной к заданной).

$А_1$) Если ресничка имеет максимальную нагрузку, то прием вещества не осуществляется.

$Б_1$) При немаксимальной нагрузке $d_1$ первой такой реснички она осуществляет прием вещества максимально возможной массы $d_2$, такой, что $d_1 + d_2 \leqslant \min(b, d)$, где $b$ - максимальная нагрузка этой реснички.

$В_1$) Следующая за ресничкой из $Б_1$) принимает массу $d_3$, как и в $Б_1$), с заменой там $d$ на $d - d_2$.

$Г_1$) Оставшаяся масса вещества опускается до следующей реснички с большим номером в ребре, для которой не выполняется условие $А_1$). Она осуществляет прием вещества по правилу $В_1$) или $Б_1$).

$Д_1$) Если ресничка в рассматриваемом ребре является последней, не удовлетворяющей условию $А_1$), то оставшаяся масса вещества делится пополам (если число нечетное, то одна из частей на единицу больше другой); и каждая из частей вещества воспринимается соответствующими ребрами, как описано выше.

$Е_1$) Процесс, описываемый позициями $А_1$)–$Д_1$), начинается с ребра, которое инцидентно корню.

*Переброс ресничкой вещества* осуществляется на следующую ресничку с меньшим номером внутри ребра по такому правилу.

$А_2$) Если следующая ресничка имеет не нулевую нагрузку, то переброс с реснички не осуществляется.

$Б_2$) Если нагрузка реснички не превосходит $r$, где $r$ - ее мера переброса, и не выполнено условие $А_2$), то перебрасывается на следующую вся нагрузка реснички и считается, что ее нагрузка становится равной нулю.

$В_2$) Если на ресничке нагрузка $m$ и $m > r$, то она перебрасывает на следующую ресничку нагрузку $r$ и оставляет у себя нагрузку $m - r$.

Если ресничка в ребре последняя, то переброс нагрузки осуществляется по правилам $А_2$), $Б_2$), $В_2$).

$Г_2$) Если ребро инцидентно корню, то переброс с наименьшей по номеру реснички осуществляется в среду по правилам $Б_2$) и $В_2$) в предположении, что среда играет роль реснички с нулевой нагрузкой.

Д$_2$) Если ребро не инцидентно корню, то есть его вершина инцидентна следующему ребру, то нагрузка с наименьшей по номеру реснички этого ребра передается наибольшей по номеру ресничке другого ребра по правилам А$_2$), Б$_2$), В$_2$).

Считаем, что процесс дыхания осуществляется в дискретные моменты времени $t = 1, 2, 3, \ldots$

В первый момент $I$-дерево $D^{-1}(b, r, n, l; V')$ имеет заданное распределение нагрузок по его ресничкам.

Ко второму моменту осуществляется прием вещества массой $d(1)$ по правилам А$_1$)–Е$_1$), и затем осуществляется переброс нагрузок с реснички на ресничку во всем $I$-дереве или выброс в среду в соответствии с правилами А$_2$), Б$_2$), В$_2$), Г$_2$), Д$_2$). А если в легкие подается масса $d$, не превосходящая объема легких, то та ее часть, которая не осела на ресничках, выбрасывается в среду.

Другими словами, за один момент (шаг) происходит «вдох» и «выдох».

Если в каждый момент $t = 1, 2, 3, \ldots$ все реснички I-дерева $D^{-1}(b, r, n, l; V')$ осуществляют прием вещества нулевой массы, то такой процесс называется *процессом самоочищения* этого I-дерева. Процесс самоочищения заканчивается в такой момент $t$, в котором нагрузки всех ресничек I-дерева $D^{-1}(b, r, n, l; V')$ впервые стали равными нулю.

Под распределением нагрузки $V'$ I-дерева $D^{-1}(b, r, n, l; V')$ будем понимать любое из возможных распределений нагрузок всех его ресничек таких, что суммарный объем их нагрузок равен $V'$. Ясно, что $V' \le V$, где $V$ - объем I-дерева $D^{-1}(b, r, n, l; V')$ и $V = 2^{l-1}bnl$. Такие распределения будем называть *конфигурациями* нагрузки $V'$ по ресничкам I-дерева $D^{-1}(b, r, n, l)$.

Занумеруем все реснички I-дерева $D^{-1}(b, r, n, l)$ таким образом, что ресничка с номером $ijk$ является $k$-ой ресничкой $j$-го ребра глубины $i$, где $1 \le i \le l$, $1 \le j \le 2^{i-1}$, $1 \le k \le n$, а нумерация ребер одной глубины идет слева направо. Тогда в каждый момент $t$ конфигурацию нагрузки $V'(t)$ в I-дереве $D^{-1}(b, r, n, l)$ можно задать набором

$$q(t) = (q_{111}(t), q_{112}(t), \ldots, q_{ijk}(t), \ldots, q_{l2^{l-1}n}(t)),$$

в котором каждая координата $q_{ijk}(t)$ равна нагрузке реснички с номером $ijk$ в момент $t$, причем $0 \le q_{ijk}(t) \le 2^{l-i}b$ и $\sum_{111}^{l2^{l-1}n} q_{ijk}(t) =$

$V'(t)$.

Пусть в процессе самоочищения конфигурации нагрузки $V'(t)$ в каждый момент $t$ изменяются по правилам $A_2) - Д_2)$. Тогда процесс самоочищения можно представить некоторым инициальным конечным автоматом без выхода с одним финальным состоянием, что было сделано автором ранее в статье [2]. Там построен такой автомат и представлена схематическая диаграмма Мура этого автомата. Состояниями построенного автомата являются конфигурации нагрузки $V'(t)$ в I-дереве $D^{-1}(b,r,n,l)$. Такие конфигурации нагрузок I-дерева $D^{-1}(b,r,n,l)$ будем называть далее состояниями этого I-дерева.

Обозначим множество состояний I-дерева $D^{-1}(b,r,n,l)$ при всевозможных его нагрузках $V'(t)$ через $Q(b,n,l)$.

Введем понятие состояния конденсации для I-дерева $D^{-1}(b,r,n,l)$. Именно, $q$ из $Q(b,n,l)$ считаем *состоянием конденсации* для I-дерева $D^{-1}(b,r,n,l)$, если в него за один шаг переходит наибольшее число состояний из $Q(b,n,l)$, то есть состояние $q$ имеет наибольшее число предшественников (прообразов).

Нашими задачами будут выяснение того, какие состояния $q$ из $Q(b,n,l)$ являются состояниями конденсации для I-дерева $D^{-1}(b,r,n,l)$, нахождение их количества, а также нахождение числа прообразов состояний конденсации.

Для решения этих задач выделим некоторые свойства состояний, которые назовем *$c_i$-свойствами* при $i = 1,2,3,4,5$.

Отметим, что $c_1$-свойство будет определено только при $b \geq 2r$, $c_2$- и $c_3$-свойства - только при $r < b < 2r$, а $c_4$- и $c_5$-свойства - только при $b = r$.

Будем говорить, что состояние $q$ из $Q(b,n,l)$ обладает:

- *$c_1$-свойством*, если при данном $q$ выполнено: $q_{ij1} = 0$, $q_{ijn} = 2^{l-(i+1)}r$, $q_{ijk} = 2^{l-i}r$, где $1 \leq i < l$, $1 \leq j \leq 2^{i-1}$, $2 \leq k \leq n-1$; при $i = l$ выполнено $q_{lj1} = 0$, $q_{ljk} = r$, где $1 \leq j \leq 2^{l-1}$, $2 \leq k \leq n-2$, и либо $q_{lj(n-1)} = r$ и $0 \leq q_{ljn} \leq b-r$, либо $1 \leq q_{lj(n-1)} \leq r-1$ и $q_{ljn} = 0$;

- *$c_2$-свойством*, если $n$-четное и при данном $q$ выполнено: $q_{ijn} = 2^{l-(i+1)}r$, $q_{ij1} = 0$, $q_{ij(2k)} = 2^{l-i}r$ и $1 \leq q_{ij(2k+1)} \leq 2^{l-i}(b-r)$, где $1 \leq i < l$, $1 \leq j \leq 2^{i-1}$, $1 \leq k \leq \frac{n-2}{2}$; при $i = l$ выполнено $q_{lj1} = 0$, $q_{ljn} = 0$, $1 \leq q_{lj(2k+1)} \leq b-r$ и $q_{lj(2k)} = r$, где $1 \leq j \leq 2^{l-1}$, $1 \leq k \leq \frac{n-2}{2}$;

- $c_3$-*свойством*, если $n$-нечетное и при данном $q$ выполнено: при $i$-четном $q_{ijn} = 2^{l-(i+1)}r$, $1 \le q_{ij(2k)} \le 2^{l-i}(b-r)$ и $q_{ij(2k-1)} = 2^{l-i}r$ и при $i$-нечетном $q_{ij1} = 0$, $1 \le q_{ij(2k+1)} \le 2^{l-i}b$ и $q_{ij(2k)} = 2^{l-i}r$, $1 \le i < l$, $1 \le j \le 2^{i-1}$, $1 \le k \le \frac{n-1}{2}$; при $i = l$, когда $l$-четно, выполнено $q_{ljn} = 0$, $1 \le q_{lj(2k)} \le b-r$ и $q_{lj(2k-1)} = r$, где $1 \le j \le 2^{l-1}$, $1 \le k \le \frac{n-1}{2}$; при $i = l$, когда $l$-нечетно, выполнено $q_{lj1} = 0$, $q_{lj2} = r$, $0 \le q_{ljn} \le b-r$, $1 \le q_{lj(2k-1)} \le b-r$ и $q_{lj(2k)} = r$, где $1 \le j \le 2^{l-1}$, $1 < k \le \frac{n-1}{2}$;

- $c_4$-*свойством*, если $n$-четное и при данном $q$ выполнено: $q_{ij(n-1)} = 0$, $q_{ijn} = 2^{l-(i+1)}b$, $1 \le q_{ij(2k)} \le 2^{l-i}b$ и $q_{ij(2k-1)} = 0$, где $1 \le i < l$, $1 \le j \le 2^{i-1}$, $1 \le k \le \frac{n-2}{2}$; при $i = l$ выполнено $q_{lj(n-1)} = 0$, $q_{ljn} = 0$, $1 \le q_{lj(2k)} \le b$ и $q_{lj(2k-1)} = 0$, где $1 \le j \le 2^{l-1}$, $1 \le k \le \frac{n-2}{2}$;

- $c_5$-*свойством*, если $n$-нечетное и при данном $q$ выполнено: при $i$-четном $q_{ijn} = 2^{l-(i+1)}b$, $1 \le q_{ij(2k-1)} \le 2^{l-i}b$ и $q_{ij(2k)} = 0$ и при $i$-нечетном $q_{ijn} = 0$, $1 \le q_{ij(2k)} \le 2^{l-i}b$ и $q_{ij(2k-1)} = 0$, где $1 \le i < l$, если $l$ - четно и $1 \le i \le l$, если $l$ - нечетно, $1 \le j \le 2^{i-1}$, $1 \le k \le \frac{n-1}{2}$; при $i = l$, когда $l$ - четно, выполнено $q_{ljn} = 0$, $1 \le q_{lj(2k-1)} \le b$ и $q_{lj(2k)} = 0$, где $1 \le j \le 2^{l-1}$, $1 \le k \le \frac{n-1}{2}$.

Пусть $C = \{c_1, c_2, c_3, c_4, c_5\}$. Класс всех состояний $q$ из $Q(b,n,l)$ с $c_i$-свойством при $c_i \in C$ обозначим через $K_{c_i}$, мощность множества $K_{c_i}$ – через $|K_{c_i}|$, а число прообразов каждого состояния $q$ из $K_{c_i}$ обозначим через $S_{K_{c_i}}$.

Справедливо следующее утверждение.

**Теорема 1.** *Множество состояний конденсации I-дерева* $D^{-1}(b,r,n,l)$ *совпадает с:*

*а)* $K_{c_1}$, *если* $b \ge 2r$;

*б)* $K_{c_2}$ *при четном* $n$ *и с* $K_{c_3}$ *при нечетном* $n$, *если* $r < b < 2r$;

*в)* $K_{c_4}$ *при четном* $n$ *и с* $K_{c_5}$ *при нечетном* $n$, *если* $b = r$.

Следующее утверждение дает решение задачи нахождения количества всех состояний конденсации I-дерева $D^{-1}(b,r,n,l)$.

**Теорема 2.** *Справедливы следующие равенства:*

*1)* $|K_{c_1}| = b^{2^{l-1}}$,

*2)* $|K_{c_2}| = \left((b-r)^{2^l-1} \cdot 2^{2^l-l-1}\right)^{\frac{n}{2}-1}$,

*3)* $|K_{c_3}| = \begin{cases} \left((b-r)^{2^l-1} \cdot 2^{2^l-l-1}\right)^{\frac{n-1}{2}}, & \text{если } l \text{ - четно}, \\ (b-r)^{2^{l-1}(n-2)-\frac{n-1}{2}} \cdot (b-r+1)^{2^{l-1}} \cdot 2^{\frac{n-1}{2}(2^l-l-1)}, & \text{иначе}, \end{cases}$

*4)* $|K_{c_4}| = (b^{2^l-1} \cdot 2^{2^l-l-1})^{\frac{n}{2}-1}$,

*5)* $|K_{c_5}| = (b^{2^l-1} \cdot 2^{2^l-l-1})^{\frac{n-1}{2}}$.

**Следствие.** *Имеет место:*

*1)* $|K_{c_2}| \asymp 2^{(2^l \cdot c_2 - l)\cdot(\frac{n}{2}-1)}$, *где* $c_2 = 1 + \log_2(b-r)$,

*2)* $|K_{c_3}| \asymp \begin{cases} 2^{(2^l \cdot c_2 - l)\cdot \frac{n-1}{2}}, & \text{если } l \text{ - четно,} \\ 2^{2^{l-1}\cdot c_3 - \frac{n-1}{2}\cdot l}, & \text{если } l \text{ - нечетно,} \end{cases}$
*где* $c_3 = n - 1 + (n-2)\log_2(b-r) + \log_2(b-r+1)$,

*3)* $|K_{c_4}| \asymp 2^{(2^l \cdot c_4 - l)\cdot(\frac{n}{2}-1)}$, *где* $c_4 = 1 + \log_2 b$,

*4)* $|K_{c_5}| \asymp 2^{(2^l \cdot c_4 - l)\cdot \frac{n-1}{2}}$

*при* $l \to \infty$.

Теперь рассмотрим задачу нахождения числа прообразов $S_{K_{c_i}}$ состояния конденсации $q$ из $K_{c_i}$ для всех $c_i \in C$.

Пусть $d'_1 = \log_2\left(2\cdot r\cdot\left(\frac{(1+\sqrt{5})^{n-3}-(1-\sqrt{5})^{n-3}}{2^{n-3}\sqrt{5}}\right)^2\right)$, $d''_1 = \log_2\left(6\sqrt{3}\cdot r\cdot\left(\frac{(1+\sqrt{5})^{n}-(1-\sqrt{5})^{n}}{2^{n}\sqrt{5}}\right)^2\right)$, $d'_2 = n - 3 + \log_2 r$, $d''_2 = n + 1 + \log_2(3\sqrt{3}\cdot r)$, $d'_3 = \frac{3n+2}{9} + \frac{1}{3}\cdot\log_2 r$, $d''_3 = \frac{9n+17}{9} + \frac{1}{3}\cdot\log_2(3\cdot r)$, $p'_3 = \frac{9n-23}{9} + \frac{2}{3}\cdot\log_2 r$, $p''_3 = \frac{9n+2}{9} + \frac{2}{3}\cdot\log_2(3\cdot r)$, $d'_4 = \log_2\left(2\cdot r\cdot\left(\frac{(1+\sqrt{5})^{\frac{n}{2}}-(1-\sqrt{5})^{\frac{n}{2}}}{2^{\frac{n}{2}}\sqrt{5}}\right)^2\right)$, $d''_4 = \log_2\left(6\sqrt{3}\cdot r\cdot\left(\frac{(1+\sqrt{5})^{\frac{n}{2}+1}-(1-\sqrt{5})^{\frac{n}{2}+1}}{2^{\frac{n}{2}+1}\sqrt{5}}\right)^2\right)$, $d'_5 = \frac{5}{9} + \frac{1}{3}\cdot\log_2 r + \frac{2}{3}\cdot\log_2\left(\frac{(1+\sqrt{5})^{\frac{n-3}{2}}-(1-\sqrt{5})^{\frac{n-3}{2}}}{2^{\frac{n-3}{2}}\sqrt{5}}\right)$, $d''_5 = \frac{14}{9} + \frac{1}{3}\cdot\log_2(27\cdot r) + 2\cdot\log_2\left(\frac{(1+\sqrt{5})^{\frac{n+1}{2}}-(1-\sqrt{5})^{\frac{n+1}{2}}}{2^{\frac{n+1}{2}}\sqrt{5}}\right)$, $p'_5 = \frac{2}{9} + \frac{1}{3}\cdot\log_2 r + \log_2\left(\frac{(1+\sqrt{5})^{\frac{n-3}{2}}-(1-\sqrt{5})^{\frac{n-3}{2}}}{2^{\frac{n-3}{2}}\sqrt{5}}\right)$, $p''_5 = \frac{2}{9} + \frac{1}{3}\cdot\log_2(27\cdot r) + \log_2\left(\frac{(1+\sqrt{5})^{\frac{n+1}{2}}-(1-\sqrt{5})^{\frac{n+1}{2}}}{2^{\frac{n+1}{2}}\sqrt{5}}\right)$.

**Теорема 3.** *Имеет место:*

*1)* $2^{2^{l-1}\cdot d'_1} \preccurlyeq S_{K_{c_1}} \preccurlyeq 2^{2^{l-1}\cdot d''_1}$,

*2)* $2^{2^{l-1}\cdot d'_2} \preccurlyeq S_{K_{c_2}} \preccurlyeq 2^{2^{l-1}\cdot d''_2}$,

*3)* $2^{2^{l-1}\cdot min(d'_3,p'_3)+\frac{1}{3}\cdot l} \preccurlyeq S_{K_{c_3}} \preccurlyeq 2^{2^{l-1}\cdot max(d''_3,p''_3)+\frac{1}{3}\cdot l}$,

*4)* $2^{2^{l-1}\cdot d'_4} \preccurlyeq S_{K_{c_4}} \preccurlyeq 2^{2^{l-1}\cdot d''_4}$,

*5)* $2^{2^{l-1}\cdot min(d'_5,p'_5)+\frac{1}{3}\cdot l} \preccurlyeq S_{K_{c_5}} \preccurlyeq 2^{2^{l-1}\cdot max(d''_5,p''_5)+\frac{1}{3}\cdot l}$

*при* $l \to \infty$.

**Следствие.** *Имеет место* $log_2 S_{K_{c_i}} \asymp 2^l$ *при* $l \to \infty$ *для всех* $i = 1, 2, 3, 4, 5$.

Автор выражает глубокую благодарность академикам Кудрявцеву Валерию Борисовичу и Чучалину Александру Григорьевичу за постановку задачи и научное руководство.

**Литература**

1. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов // — М.: Наука, 1985. — 320 с.

2. Гераськина Ю. Г. Об одной автоматной модели в биологии // Дискретная математика — 2007. — Т. 19, вып. 3, — С. 122–139.

3. Гераськина Ю. Г. Модель процесса дыхания живых организмов // Интеллектуальные системы — 2003. — Т. 8, вып. 1-4, — С. 429–456.

4. Гераськина Ю. Г. Модель самоочищения легочных структур // Интеллектуальные системы — 2002-2003. — Т. 7, вып. 1-4, — С. 41–54.

5. Прудников А. П., Брычков Ю. А., Маричев О. И. Интегралы и ряды // — М.: Наука, 1981. — 800 с.

## ON SOME TYPES OF AUTOMATA OVER FINITE RING

**Skobelev V.G. (Ukraine, Donetsk, IAMM of NAS of Ukraine)**
*skbv@iamm.ac.donetsk.ua*

Applications of the ring theory in the process of design of modern ciphers has grounded actuality of investigation of automata presented via systems of equations over finite ring [1,2]. Some basic characteristics of general models of Mealy and Moore automata over arbitrary finite associative-commutative ring $\mathcal{K} = (K, +, \cdot)$ with the unit are presented in the given paper.

We denote by $\mathcal{A}_{n,1}$ the set of all Mealy automata

$$\mathbf{q}_{t+1} = \mathbf{f}_1(\mathbf{q}_t) + \mathbf{f}_3(\mathbf{x}_{t+1}),$$

$$\mathbf{y}_{t+1} = \mathbf{f}_2(\mathbf{q}_t) + \mathbf{f}_4(\mathbf{x}_{t+1})$$

and by $\mathcal{A}_{n,2}$ the set of all Moore automata

$$\mathbf{q}_{t+1} = \mathbf{f}_1(\mathbf{q}_t) + \mathbf{f}_3(\mathbf{x}_{t+1}),$$

$$\mathbf{y}_{t+1} = \mathbf{f}_2(\mathbf{q}_{t+1}),$$

where $\mathbf{f}_i : K^n \to K^n$ $(i = 1, \ldots, 4)$ ($\mathbf{f}_2$ is a non-linear mapping) and $\mathbf{q}_t, \mathbf{x}_t, \mathbf{y}_t \in K^n$ are correspondingly internal state, input and output at instant $t \in \mathbf{Z}_+$.

Let $\mathcal{A}_{n,i}^{inv}$ $(i = 1, 2)$ be the set of all automata $M_i \in \mathcal{A}_{n,i}$ such that for every initial state $\mathbf{q}_0 \in K^n$ the mapping $\mathbf{F}_{(M,\mathbf{q}_0)} : (K^n)^+ \to (K^n)^+$, realized by initialized automaton $(M, \mathbf{q}_0)$ is bijection. It is evident that

$$\mathcal{A}_{n,1}^{inv} = \{M_1 \in \mathcal{A}_{n,1} | \mathbf{f}_4 : K^n \to K^n \ \textit{is bijection}\},$$

$$\mathcal{A}_{n,2}^{inv} = \{M_2 \in \mathcal{A}_{n,2} | \mathbf{f}_2 : K^n \to K^n \ \textit{and} \ \mathbf{f}_3 : K^n \to K^n \ \textit{are bijections}\}.$$

It is worth to note that the inverse $M_i^{-1}$ $(i = 1, 2)$ of an automaton $M \in \mathcal{A}_{n,i}^{inv}$ is Mealy automaton.

Automata $M_i \in \mathcal{A}_{n,i}^{inv}$ $(i = 1, 2)$ determine the class of stream ciphers for which initial state is secret short-term key, while parameters are long-term key. For any stream cipher $((M, \mathbf{q}_0), (M^{-1}, \mathbf{q}_0))$ $(M \in \mathcal{A}_{n,1}^{inv} \cup \mathcal{A}_{n,2}^{inv})$ in the process "coding-decoding"automata $M_i$ and $M_i^{-1}$ move in the space of states during the same trajectory in the same direction.

Some non-trivial subsets of the sets $\mathcal{A}_{n,i}$ $(i = 1, 2)$ can be characterized in the following way:

1) an automaton $M \in \mathcal{A}_{n,1} \cup \mathcal{A}_{n,2}$ is a strongly connected one if and only if $\mathbf{f}_3 : K^n \to K^n$ is bijection;

2) if $\mathbf{f}_3 : K^n \to K^n$ is bijection then an automaton $M \in \mathcal{A}_{n,1}$ is a reduced one and any of its two states can be distinguished by any input symbol;

3) if $\mathbf{f}_1 : K^n \to K^n$ and $\mathbf{f}_2 : K^n \to K^n$ are bijections then an automaton $M \in \mathcal{A}_{n,2}$ is a reduced one and any of its two states can be distinguished by any input symbol;

4) states $\mathbf{q}, \widetilde{\mathbf{q}} \in K^n$ $(\mathbf{q} \neq \widetilde{\mathbf{q}})$ of an automaton $M \in \mathcal{A}_{n,1} \cup \mathcal{A}_{n,2}$ are twins if and only if they are elements of the same class of the partition $K^n/\varepsilon$ where $\varepsilon = \ker \mathbf{f}_1 \cap \ker \mathbf{f}_2$ for $M \in \mathcal{A}_{n,1}$ and $\varepsilon = \ker \mathbf{f}_1$ for $M \in \mathcal{A}_{n,2}$.

Let the subset $\widetilde{\mathcal{A}}_{n,1}$ of the set $\mathcal{A}_{n,1}$ consists of all automata

$$\mathbf{q}_{t+1} = A\mathbf{q}_t\mathbf{q}_t^T\mathbf{b} + C\mathbf{q}_t + \mathbf{d} + E\mathbf{x}_{t+1},$$

$$\mathbf{y}_{t+1} = G\mathbf{q}_t + F\mathbf{x}_{t+1}$$

and the subset $\widetilde{\mathcal{A}}_{n,2}$ of the set $\mathcal{A}_{n,2}$ consists of all automata

$$\mathbf{q}_{t+1} = A\mathbf{q}_t\mathbf{q}_t^T\mathbf{b} + C\mathbf{q}_t + \mathbf{d} + E\mathbf{x}_{t+1},$$

$$\mathbf{y}_{t+1} = G\mathbf{q}_{t+1}$$

where $\mathbf{b} = (b^{(1)}, \ldots, b^{(n)})^T \in K^n$, $\mathbf{d} = (d^{(1)}, \ldots, d^{(n)})^T \in K^n$ and $A, C, E, G, F$ are $n \times n$-matrices.

Complexity of identification of initial state $\mathbf{q}_0 \in K^n$ of an automaton $M \in \widetilde{\mathcal{A}}_{n,1} \cup \widetilde{\mathcal{A}}_{n,2}$ can be characterized in the following way.

It is supposed that the experimenter can apply any experiment of any multiplicity. The problem of identification of initial state of an automaton $M \in \widetilde{\mathcal{A}}_{n,1}$ is trivial, if $G$ is an invertible matrix. In all other cases this problem is a hard one. Its high complexity is justified by the following reasons. Firstly, searching in the set of input sequences is a hard problem. Secondly, design the set of solutions for non-linear systems of equations is a hard problem. Thirdly, checking the property "to be subset of the set of equivalent states"is also a hard problem.

It is worth to note that additional condition $M \in \widetilde{\mathcal{A}}_{n,1}^{inv} \cup \widetilde{\mathcal{A}}_{n,2}^{inv}$ does not simplify the problem of identification of initial state. Thus, selection

initial state of an automaton $M \in \widetilde{\mathcal{A}}_{n,1}^{inv} \cup \widetilde{\mathcal{A}}_{n,2}^{inv}$ in the role of short-term key for corresponding stream cipher is grounded.

For an automaton $M \in \widetilde{\mathcal{A}}_{n,1} \cup \widetilde{\mathcal{A}}_{n,2}$ complexity of parametric identification can be characterized in the following way.

Parametric identification for an automaton $M_2 \in \widetilde{\mathcal{A}}_{n,2}$ is a hard problem, since it is always is reduced to design the set of solutions for non-linear systems of equations.

Let $M_1 \in \widetilde{\mathcal{A}}_{n,1}$. It is easy to identify matrices $G$ and $F$. Also it is easy to identify vector $\mathbf{d}$ and matrix $E$ if and only if $G$ is an invertible matrix. But it is a hard problem to identify matrices $A, C$ and vector $\mathbf{b}$ since it is always is reduced to design the set of solutions for non-linear systems of equations.

It is worth to note that additional condition $M \in \widetilde{\mathcal{A}}_{n,1}^{inv} \cup \widetilde{\mathcal{A}}_{n,2}^{inv}$ does not simplify the problem of parametric identification. Thus, selection parameters of an automaton $M \in \widetilde{\mathcal{A}}_{n,1}^{inv} \cup \widetilde{\mathcal{A}}_{n,2}^{inv}$ in the role of long-term key for corresponding stream cipher is grounded.

The set $S_{fxd}^{(i)}(M, \mathbf{q}_0)$ $(M \in \widetilde{\mathcal{A}}_{n,1} \cup \widetilde{\mathcal{A}}_{n,2}, \mathbf{q}_0 \in K^n)$ of fixed points of the length $i$ for the mapping $\mathbf{F}_{(M,\mathbf{q}_0)}$ can be characterized in the following way ($I$ is the unit matrix):

1) $S_{fxd}^{(1)}(M_1, \mathbf{q}_0) \neq \emptyset$ $(M_1 \in \widetilde{\mathcal{A}}_{n,1})$ if and only if the set of solutions of equation $(I - F)\mathbf{x} = G\mathbf{q}_0$ is not empty (thus, if the matrix $I - F$ is an invertible one then $|S_{fxd}^{(i)}(M_1, \mathbf{q}_0)| = 1$ $(i \in \mathbf{N}, \mathbf{q}_0 \in K^n)$);

2) if $F = I$ then $S_{fxd}^{(1)}(M_1, \mathbf{q}_0) = K^n$ $(M_1 \in \widetilde{\mathcal{A}}_{n,1})$ for any initial state $\mathbf{q}_0 \in K^n$ such that $G\mathbf{q}_0 = \mathbf{0}$;

3) $S_{fxd}^{(1)}(M_2, \mathbf{q}_0) \neq \emptyset$ $(M_2 \in \widetilde{\mathcal{A}}_{n,2})$ if and only if the set of solutions of equation $(G^{-1} - E)\mathbf{x} = A\mathbf{q}_0\mathbf{q}_0^T B + C\mathbf{q}_0 + \mathbf{d}$ is not empty (thus, if the matrix $G^{-1} - E$ is an invertible one then $|S_{fxd}^{(i)}(M_2, \mathbf{q}_0)| = 1$ $(i \in \mathbf{N}, \mathbf{q}_0 \in K^n)$);

4) if $E = G^{-1}$ then $S_{fxd}^{(1)}(M_2, \mathbf{q}_0) = K^n$ $(M_2 \in \widetilde{\mathcal{A}}_{n,2})$ for any initial state $\mathbf{q}_0 \in K^n$ such that $A\mathbf{q}_0\mathbf{q}_0^T B + C\mathbf{q}_0 + \mathbf{d} = \mathbf{0}$.

## References

1. V.V. Skobelev, V.G. Skobelev. Analysis of ciphersystems. — Donetsk: IAMM of NAS of Ukraine, 2009 – 479 p.

2. V.V. Skobelev, N.M. Glazunov, V.G. Skobelev. Manifolds over rings. Theory and applications. — Donetsk: IAMM of NAS of Ukraine, 2011. – 323 p.

# Секция “Защита информации”

# МНОГОПОТОЧНЫЕ СЕРВЕРА, ИСПОЛЬЗУЮЩИЕ ОБРАБОТЧИКИ СОБЫТИЙ

**Александров Д.Е. (Механико-математический факультет МГУ им. М.В.Ломоносова)**
*d06alexandrov@gmail.com*

Исследуются основные архитектуры TCP серверов, реализуемых на SMP-системах. Выделены параметры архитектурных решений, характеризующие производительность серверов. На основании полученных данных разработан сервер с архитектурой, превосходящей по производительности существующие архитектуры.

**Введение**

Один из основных вопросов при выборе сервера для реализации какого-либо протокола — баланс между производительностью обработки запросов и низкими затратами при портировании готовых программных решений на базу интернет-сервера. Использование обработчиков событий позволяет существенно сократить расходы на обработку группы соединений, что позволяет эффективно использовать кэширование часто используемых данных.

Для анализа интернет-серверов на базе SMP-систем были выделены основные параметры, позволяющие сделать сравнительные оценки производительности архитектур. На основании полученных данных была спроектирована гибридная архитектура, превосходящая по этим характеристикам существующие архитектуры.

**Серверные архитектуры**

Основными серверными архитектурами являются:

- **Однопоточная событийно-зависимая**. *Squid proxy* [4].
- **Многопроцессная**. *Apache web-server (MPM worker module)* [5]
- **Многопоточная**. *Apache web-server* [5]
- **Многопоточная событийно-зависимая**. *Flash web-server* [2]

При изучении архитектур были выделены следующие параметры, влияющие на производительность:

- **Количество одновременно обрабатываемых соединений**.

- **Переключение контекста**.
- **Адресное пространство**.
- **Портируемость кода**.

**Гибридная архитектура**

Основой этой архитектуры является главный поток, синхронизирующий деятельность рабочих потоков. Рабочие потоки — потоки, осуществляющие непосредственную обработку соединения путем переключения контекста с потока на обработчик.

В самом начале работы сервера создается набор пустых обработчиков соединений (одно соединение — один обработчик), являющихся облегченными версиями стандартных потоков. Каждый обработчик имеет свой стек вызовов и хранит основную информацию о соединении. Соединению, создаваемому при подключении к серверу, ставится в соответствие свободный обработчик. Далее главный поток проверяет, есть ли свободные рабочие потоки или нет. Если нет, то обработчик ставится в очередь. В противном случае к одному из свободных рабочих потоков отправляется соответствующий запрос на исполнение кода обработчика. Когда запрос с информацией об обработчике приходит, рабочий поток меняет свой контекст выполнения на контекст обработчика и начинает (продолжает) выполнение кода обработки соединения. Когда требуется получить данные с неблокирующего сокета, соответствующее сообщение отправляется главному потоку, и рабочий поток ожидает следующей команды, выходя из контекста текущего обработчика, а главный поток добавляет в свой список ожидаемых событий событие, затребованное обработчиком.

**Результаты общего сравнения**

По результатам теоретического сравнения основных параметров различных архитектур можно сделать вывод о преимуществе гибридной архитектуры. Во-первых, количество переключений контекстов потоков сокращается до минимума по сравнению с многопроцессными и многопоточными серверами. Кроме того, все рабочие потоки действуют в рамках одного адресного пространства, позволяя организовывать различные механизмы кэширования. Во-вторых, имеется возможность портировать готовый исходный код под гибридную архитектуру без больших затрат, в отличие от событийно-зависимых архитектур.

**Особенности реализации**

Количество одновременно обрабатываемых соединений физически не может превышать количество доступных ядер процессоров. В случае, если задать количество рабочих потоков приблизительно равным количеству ядер процессоров, мы получаем максимум количества параллельно обрабатываемых соединений.

Важнейшей частью гибридной архитектуры является реализация своего переключения контекста. Так как на высоконагруженных серверах количество ядер/процессоров всегда будет на несколько порядков меньше, чем количество соединений, то необходимо максимально снизить затраты на смену контекста. В стандартной реализации потоков в `Linux` помимо непосредственно переключения контекста (сохранение/загрузка некоторых системных регистров, в том числе регистров стека вызовов), производится довольно большое количество дополнительных действий. Нам же достаточно реализовать функции смены контекста, при использовании которых будут восстанавливаться стек и часть регистров согласно `AMD64 ABI` [1, ch.3.2.1] (стандарт реализации функций 64х разрядных систем). В рамках такого подхода были реализованы несколько основных функций: `code_start()` — инициализация стека обработчика и запуск функции обработки соединения, `code_pause()` — приостановка выполнения кода обработчика с возможностью продолжить при вызове команды `switch_context()`.

Благодаря тому, что у нас имеется только одно адресное пространство и затраты на процесс «общения» между рабочими и главным поток малы, мы имеем возможность кэшировать часто запрашиваемые данные. Главный поток хранит информацию об открытых файлах, которые были отображены на память процесса с помощью системного вызова `mmap()`. Когда рабочему потоку требуется открыть или прочитать информацию из файла, он «спрашивает» у главного потока открыт ли данный файл. Если данный файл открыт и отображен на память, то чтение производится из кэша. В противном случае самостоятельно открывает файл и в зависимости от настроек сервера и объема свободной оперативной памяти может сам осуществить кэширование нового файла.

## Тестирование

Для проведения тестов был реализован простой веб-сервер (разбор простого `GET`-запроса и отправка файла в качестве ответа). Благодаря портируемости данный код был использован и на сервере с гибридной архитектурой, и на сервере с многопоточной архитектурой.

Тестирование показало, что количество запущенных потоков действительно играет важную роль в производительности сервера. Сервер с гибридной архитектурой (4 потока) имел реальную скорость передачи данных — около 70МБ/с (погрешность вычислений не больше 1%). Многопоточный же сервер (200 потоков) имел скорость около 65МБ/с. Таким образом разница составила около $5 - 8\%$. Кроме того исследования показали, что увеличение рабочих потоков гибридной системы (при неизменности доступных ядер процессора) приводит к снижению производительности.

## Заключение

Результатом исследования архитектур, используемых на данный момент, стала предложенная гибридная архитектура, сочетающая в себе преимущества событийно-зависимых серверов и многопоточных серверов. Когда объем свободной памяти допускает — используются механизмы первого типа серверов. При большой нагрузке — работа сервера схожа с работой многопоточных серверов, однако превосходит по производительности, т.к. минимизировано громоздкое переключение контекста. Кроме того, благодаря низкоуровневому программированию, стало возможным портировать уже реализованные алгоритмы на данный сервер без дополнительных трудозатрат.

Выдвинутые теоретические предположения были подтверждены тестированием разработанного сервера.

Автор выражает благодарность своему научному руководителю Панкратьеву Антону Евгеньевичу и Галатенко Алексею Владимировичу за общее руководство и помощь в проведении тестирования разработанного сервера.

**Литература**

1. M. Matz, J. Hubicka, A. Jaeger, and M. Mitchell. System V Application Binary Interface AMD64 Architecture Processor Supplement 0.99.5 // September 3, 2010

2. N. Zeldovich. Concurrency Control for Multi-Processor Event-Driven Systems // Massachusetts Institute of Technology, June, 2002

3. D. P. Bovet, M. Cesati. Understanding the Linux Kernel // O'Reilly Media, October, 2000

4. http://www.squid-cache.org

5. http://httpd.apache.org

# МЕТОДЫ ОПТИМИЗАЦИИ ГЛУБИНЫ РЕАЛИЗАЦИИ ХЭШ-ФУНКЦИЙ

**Болотов А.А., Галатенко А.В., Гринчук М.И., Золотых А.А., Иванович Л.**

*agalat@msu.ru*

**Введение.** В работе предлагается ряд методов оптимизации схемной реализации арифметических алгоритмов хэширования: семейства SHA-2 ([1]) и функций SHA-1 ([1]) и MD5 ([2]). Рассматриваемые алгоритмы хэширования состоят из двух этапов — предобработки и собственно выработка хэша. Предобработка включает в себя выравнивание сообщения, разрезание на блоки и инициализацию значений. На этом этапе трудоемких вычислений не производится. Выработка хэша заключается в итеративной обработке блоков сообщения; именно эта часть представляет основную сложность.

В ряде приложений (таких как высокоскоростные сети или дисковые массивы с поддержкой механизмов безопасности) требуется высокая пропускная способность используемых реализаций хэш-функций, достижимая только для аппаратных решений. Определяющим параметром производительности таких реализаций является глубина схемы. Под глубиной понимается длина максимального простого пути схемы. Вторичным параметром оптимизации является сложность, то есть общее число элементов схемы. Рассматривается базис из элементов конъюнкции, дизъюнкции, отрицания и задержки (при этом отрицание игнорируется при вычислении глубины и сложности).

Предлагаемые методы позволяют существенно понижать глубину схем, реализующих рассматриваемые алгоритмы хэширования. Применение этих методов позволяет получать схемы с глубиной, меньшей, чем у известных реализаций. Часть используемых методов явно или неявно рассматривалась в литературе ([3], [4], [5]). Однако в случае явного рассмотрения делались предположения, существенно понижающие общность и как следствие завышающие теоретические нижние оценки глубины ([3], [4]). Предлагаемые схемы имеют глубину, меньшую, чем соответствующие нижние оценки. Схема в работе [5], превосходящая реализацию в работе [3], имеет на один ярус элементов больше, чем предлагаемая реализация.

**Методы оптимизации.** Основными использованными методами оптимизации являются диагональный разрез (в работах [3] и [4]

он строился с помощью анализа графа потока данных), симплификация цепочки сумматоров, спекулятивные вычисления и перестановка сумматора и циклического сдвига. Дополнительная оптимизация была достигнута благодаря тождественным преобразованиям булевых функций и разложению булевых функций по переменной.

Диагональный разрез заключается в скашивании границ циклов (этот прием часто применяется при оптимизирующей компиляции программ). Содержательно это можно проиллюстрировать следующим образом. Если изобразить вычисление хэш-функции вертикальной полосой, то вместо стандартного разбиения на прямоугольники, соответствующие раундам преобразования, осуществляется разбиение этой полосы на параллелограммы (по сути путем смещения элементов задержки по схеме) для минимизации длины максимального пути. Возникающие при этом снизу и сверху дополнительные треугольники могут быть включены в схему регулярным образом (иными словами, дополнены до параллелограммов) за счет задания корректных входных значений при инициализации.

Симплификация цепочки сумматоров основана на следующей известной идее. Сумма $x+y+z$ может быть вычислена как сумма $A+B$, где $A = x \oplus y \oplus z$, $B = ShL^1(Maj(x, y, z))$. Это преобразование уменьшает как глубину, так и сложность схемы.

Спекулятивные вычисления также часто применяются при оптимизирующей компиляции программ. Выигрыш здесь может заключаться в том, что вычисление результата прохождения ветви и вычисление значения, определяющего выбор ветви, могут осуществляться параллельно. При схемной реализации алгоритма SHA-1 применение этого приема при вычислении функций $f_t$ позволило получить схему с глубиной, равной глубине сумматора.

Идея перестановки сумматора и циклического сдвига заключается в следующем. Вычисление $ShL^k(A + B)$ можно заменить на эквивалентное выражение $ShL^k(A) + ShL^k(B) + C$, где $C$ — один из элементов четырехэлементного множества $\mathcal{C}_k$, зависящего только от $k$. Выбор элемента определяется значением битов переноса $A+B$. За счет описанной перестановки при реализации MD5 удалось удлинить цепочку сумматоров и понизить глубину благодаря симплификации удлиненной цепочки.

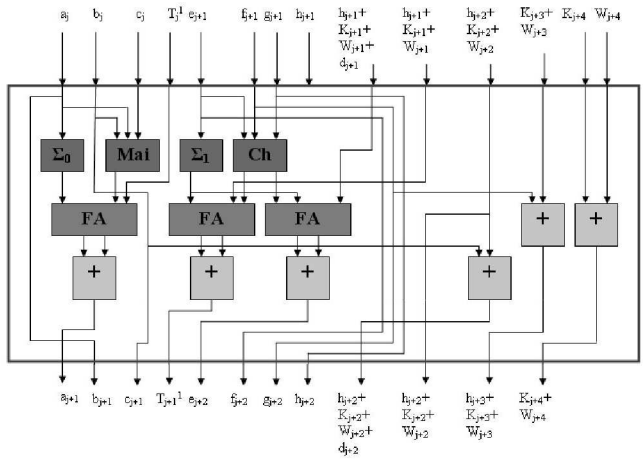



Рис. 1. Реализация алгоритмов семейства SHA-2.

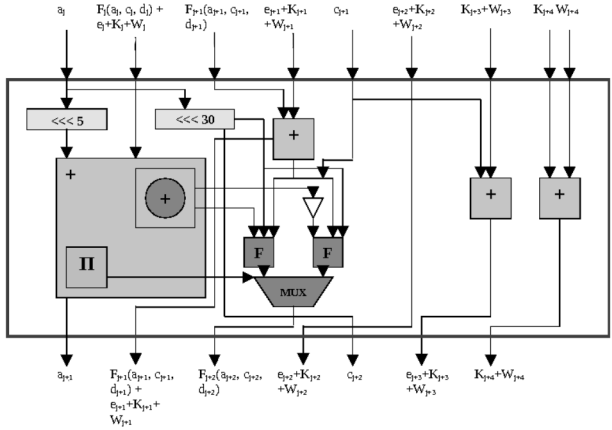



Рис. 2. Реализация алгоритма SHA-1.

**Схемные реализации.** Применение описанных методов при реализации функций семества SHA-2 позволило получить схему (рис. 1) глубиной $6 + D(n)$, где $D(n)$ — глубина сумматора соответствующей ширины (32 для SHA-224 и SHA-256, 64 для SHA-384 и SHA-512).

При этом блок Maj реализуется с глубиной 2 за счет разложения по первой переменной и предвычисления конъюнкции и дизъюнкции второго и третьего аргумента. Стандартная реализация блока Maj увеличивает приведенную оценку глубины на 1.

Применение описанных методов при реализации функции SHA-1 позволило получить схему (рис. 2) глубиной $D(32)$.

Применение описанных методов при реализации функции MD5 позволило получить схему (рис. 3) глубиной $D(32) + 10$.

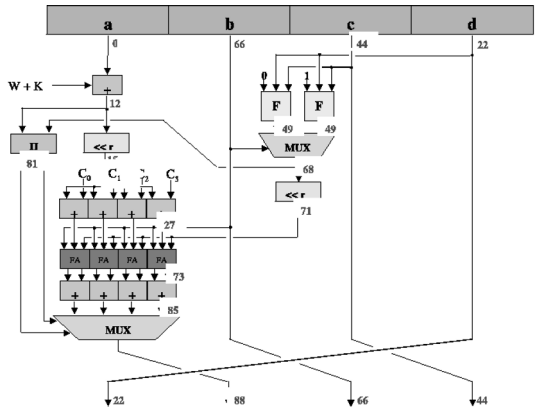



Рис. 3. Реализация алгоритма MD5.

**Литература**

1. NIST, FIPS PUB 180-2, 2001.

2. Rivest R. The MD5 Message-Digest Algorithm. — RFC 1321, 1992.

3. Lee Y. K. et al. Hardware design for Hash functions // Secure Integrated Circuits and Systems, Integrated Circuits and Systems, pp. 79–104, 2010.

4. Lee Y. K. et al. Design Methodology for Throughput Optimum Architectures of Hash Algorithms of the MD4-class // Journal of Signal Processing Systems 53(1-2), pp. 89–102, 2008.

5. Dadda L. et al. The Design of a High Speed ASIC Unit for the Hash Function SHA-256 (385, 512) // DATE'04, pp. 70–75, 2004.

## О ВОССТАНОВЛЕНИИ ПАРАМЕТРОВ $\varepsilon$-БЕЗОПАСНОСТИ

**Галатенко А.В.**
*agalat@msu.ru*

Напомним определение $\varepsilon$-безопасного языка, введенное в работе [1]. Под конечным автоматом мы будем понимать четверку $V = (A, Q, \varphi, q_0)$, где $A$ — конечное множество входных символов, $Q$ — конечное множество состояний, $\varphi : A \times Q \to Q$ — функция переходов, $q_0 \in Q$ — начальное состояние. Пусть $Q = S \cup I$, причем $S \cap I = \emptyset$. Состояния из $S$ назовем безопасными, состояния из $I$ — небезопасными. Далее будем предполагать, что начальное состояние является безопасным, все состояния достижимы из начального, а $|Q| > 1$.

Обозначим через $A^*$ множество всех конечных слов в алфавите $A$. Функция $\varphi$ может быть продолжена на множество $A^* \times Q$ по мультипликативности.

Подмножество $A^*$ называется языком. Каждому слову $\alpha \in A^*$ соответствует слово $\kappa(\alpha) \in Q^*$, $\kappa(\alpha) = \varphi(\alpha, q_0)$. Рассмотрим произвольное $\varepsilon > 0$. Введем функции $s : Q^* \to \mathbb{N} \cup \{0\}$ и $i : Q^* \to \mathbb{N} \cup \{0\}$ следующим образом. Функция $s(\kappa)$ равняется числу букв $\kappa$, содержащихся в $S$, $i(\kappa)$ равняется числу букв $\kappa$, содержащихся в $I$. Обозначим через $|\kappa|$ число букв в слове $\kappa$. Назовем слово $\kappa$ $\varepsilon$-безопасным, если $\frac{i(\kappa)}{|\kappa|} \leq \varepsilon$. Назовем язык $\mathcal{A}$ $\varepsilon$-безопасным ($S_\varepsilon$-языком), если все слова из $\mathcal{A}$ $\varepsilon$-безопасны, и не существует $\varepsilon$-безопасных слов, не принадлежащих $\mathcal{A}$. В работе [1] показано, что $\varepsilon$-безопасные языки являются контекстно-свободными и вообще говоря не регулярными; если $\varepsilon_1, \varepsilon_2 \in \mathbb{Q}$, $0 \leq \varepsilon_1 < \varepsilon_2 \leq 1$, то существует язык, являющийся $S_{\varepsilon_1}$-языком, но не являющийся $S_{\varepsilon_2}$-языком; если $\varepsilon_2$ не равно 1, то существует язык, являющийся $S_{\varepsilon_2}$-языком, но не являющийся $S_{\varepsilon_1}$-языком.

В работе [2] решалась задача восстановления $\varepsilon$-безопасного языка с помощью кратного условного эксперимента при наличии оракула, для каждого входного слова указывающего, является ли слово $\varepsilon$-безопасным. Были найдены необходимые и достаточные условия восстановления языка с помощью конечного эксперимента при известном разбиении множества состояний и неизвестном значении $\varepsilon$. В частности, если значение $\varepsilon$ таково, что в диаграмме Мура автомата имеется пара циклов, причем в одном цикле доля небезопасных состояний больше $\varepsilon$, в другом — меньше, и циклы соединены ориен-

тированным путем, восстановление $\varepsilon$-безопасного языка с помощью конечного эксперимента невозможно.

Рассмотрим обратную задачу. Пусть значение $\varepsilon$ известно, и требуется восстановить разбиение множества состояний. Имеет место следующий результат.

**Теорема.** *Для любого автомата $V$ и любого рационального значения $\varepsilon$, $0 \leq \varepsilon \leq 1$, существует конечный кратный условный эксперимент, восстанавливающий задаваемый автоматом $\varepsilon$-безопасный язык.*

Автор выражает глубокую благодарность своему научному руководителю, д.ф.-м.н., проф. В.Б. Кудрявцеву за постановку задачи и внимание к работе.

**Литература**

1. Галатенко А. В., Автоматные модели защищенных компьютерных систем // "Интеллектуальные системы", т.11, вып. 1–4, Москва, 2007, с. 403–418.

2. Галатенко А. В., О восстановлении разбиения безопасности // "Интеллектуальные системы", т.14, вып. 1–4, Москва, 2010.

# К ВОПРОСУ ПОСТРОЕНИЯ ФОРМАЛЬНЫХ МОДЕЛЕЙ СБОЕУСТОЙЧИВЫХ ПРИЛОЖЕНИЙ РЕАЛЬНОГО ВРЕМЕНИ

**Галатенко В.А., Костюхин К.А., Шмырев Н.В.**
**(Научно-исследовательский институт системных исследований РАН, Москва)**
*galat@niisi.msk.ru, kost@niisi.msk.ru, shmyrev@niisi.msk.ru*

Моделирование приложений, функционирующих в реальном масштабе времени, требует наличия формализма, позволяющего описать объекты реального времени и их взаимодействие, унифицировать подходы к описанию вычислений. В этой работе мы построим модель вычислений, описывающую основные аспекты контролируемого выполнения приложений реального времени, включающую в себя следующее:

- Статическую и динамическую верификацию корректности
- Оптимизацию и проверку результатов оптимизации
- Устойчивость к сбоям
- Представление вычисления с различной степенью детализации
- Возможность описания ресурсов

Формальное описание устойчивой системы реального времени приводится в работе [1]. В качестве модели приложения в этой работе используются системы переходов [2], в качестве общей спецификации – логика действий со временем (TLA – Temporal Logic of Actions) [3].

Сбои моделируются с помощью множества «сбойных» действий, которые изменяют состояние так же, как и обычные вычисления. Устойчивость к сбоям обеспечивается проведением корректирующих действий.

Приведем несколько определений:

**Определение 1.** *Состояние – отображение набора переменных $Var$ на набор значений $Val$.*

$$s : Var \to Val$$

**Определение 2.** *Действие – это некоторый предикат над переменными, их значениями, а также их измененными значениями,*

*которые обозначаются штрихом:*

$$x' + 1 \leqslant y$$

Над бесконечными последовательностями действий $\sigma = \sigma_1, \sigma_2, \ldots$ можно рассматривать временные формулы, которые составлены из булевых операторов и операторов линейной временной логики. Например, $[\Box\phi](\sigma)$ – предикат $\phi$ выполняется для любого суффикса $\sigma$, $[\diamond\phi](\sigma)$ – предикат $\phi$ выполняется хотя бы однажды.

**Определение 3.** *Программа – набор, состоящий из следующих компонент:*

1. *Конечное непустое множество $\overline{v}$ состояний;*
2. *Подмножество внутренних состояний $\overline{x}$ множества $\overline{v}$;*
3. *Предикат первоначального состояния $\Theta$, включающий в себя только переменные из $\overline{v}$;*
4. *Конечный набор $A$ действий над переменными из $\overline{v}$.*

**Определение 4.** *Вычисление программы $P = (\overline{v}, \overline{x}, \Theta, A)$ – это последовательность состояний $\sigma$, такая что выполняются два условия:*

1. *$\sigma_0$ удовлетворяет $\Theta$*
2. *$\sigma_{i+1} = \sigma_i$ или найдется такое $\tau \in A$, что $\sigma_{i+1}, \sigma_i$ удовлетворяет $\tau$.*

**Определение 5.** *Для программы $P = (\overline{v}, \overline{x}, \Theta, A)$ рассмотрим*

$$N_P = \bigvee_{\tau \in A} \tau$$

*Тогда точная спецификация определяется как*

$$\Pi(P) = \Theta \vee \Box[N_P]_{\overline{v}}$$

*и описывает набор всех разрешенных состояний программы.*

**Определение 6.** *Внешняя спецификация программы задается как:*

$$\Phi(P) = \exists\overline{x}.\Pi(P)$$

*и описывает все возможные последовательности внешних состояний системы.*

И, наконец, введем определение уточнения программ, позволяющее ввести понятие верификации.

**Определение 7.** *Отношение уточнения $P_l \sqsubseteq P_h$ означает что программа $P_l$ корректно реализует $P_h$, то есть отношение*

$$\Phi(P_l) \Rightarrow \Phi(P_h)$$

*выполнено для внешних спецификаций.*

**Моделирование разных аспектов вычисления**

Естественно описывать вычисление несколькими моделями, отражающими различные аспекты приложения, например, модель памяти и модель занимаемой пропускной способности сети могут быть различными, дополняя основную модель вычислений. Кроме того, набор ограничений и утверждений о приложении может быть достаточно разнородным и задаваться:

1. результатами тестирования;
2. выведенными в процессе анализа приложения утверждениями;
3. проверками во время выполнения;

**Определение 8.** *Введем внутреннюю и внешнюю спецификацию набора программ $P_i$:*

$$\Pi(P_1, \ldots, P_n) = (\wedge \Theta_i) \vee (\wedge \Box [N_{P_i}]_{\overline{v}})$$

$$\Phi(P_1, \ldots, P_n) = \exists \overline{x}. \Pi(P_1, \ldots, P_n)$$

**Определение 9.** *Будем говорить, что программа $P$ является уточнением набора $P_1, \ldots, P_N$, если*

$$\Phi(P) \Rightarrow \Phi(P_1, \ldots, P_n)$$

Таким образом можно определить отношение частичного порядка на множестве наборов программ и использовать его для построения целого семейства моделей, представляющих выполнение приложения с разной степенью детализации.

**Моделирование ресурсов**

Так же, как и время, потребляемые ресурсы важны для доказательства корректности работы приложения и для выполнения его миссии. Для того, чтобы описать потребление ресурсов, также, как и в случае со временем необходимо ввести внутренние переменные, которые описывают ресурс и дополнить соотношения действий границами использования ресурсов.

Например, время моделируется с помощью временных меток действий. Каждое действие $\tau$ связывается с нижней временной границей $L(\tau)$ и верхней временной границей $U(\tau)$. Для того, чтобы считаться успешным, действие должно выполняться по крайней мере $L(\tau)$ временных интервалов. Действие не должно выполняться реже, чем $U(\tau)$.

Определим набор ресурсов $z_i \in Z$ и для каждого из них и для каждого действия определим верхнюю и нижнюю границу потребления ресурсов для каждого действия $U_i(\tau)$ и $L_i(\tau)$.

Для каждого из ресурсов нужно ввести и специальные ограничения, которые описывают их расходование, например, для времени:

$$now = 0 \in \Theta \tag{1}$$

$$\Box[now' \in (now, \infty))]_{now} \tag{2}$$

$$\forall t \in R^+, \diamond(now > t) \tag{3}$$

Для механизма простейшего выделения памяти формулируются более простые условия $\Box mem(\sigma) > 0, mem' > mem - U_m(\tau), mem' < mem - L_m(\tau)$, хотя, для сложных механизмов выделения можно учитывать и более точные характеристики, такие как фрагментацию памяти.

В случае работы с ресурсом нам необходимо расширить определение внутренней и внешней спецификации соответственно:

$$R_P = \bigvee_{\tau \in A} R_i(\tau)$$

$$\Pi(P) = \Theta \vee \Box[N_P]_{\overline{v}} \vee \Box[R_P]_{\overline{v}}$$

где $R_i$ – ограничения, накладываемые на использование ресурса.

**Оптимизация**

Возможность учета понятия оптимизации является естественным требованием к формальной модели приложения. Необходимо рассмотреть две проблемы, возникающие при этом - проблему проверки корректности оптимизирующего преобразования и проблему количественного определения результатов оптимизации. Первая проблема может быть решена с помощью введения эталонной модели вычисления $P_e$, таким образом корректность оптимизации можно определить как соответствие эталонной модели вычисления $P_1 \sqsubseteq P_e \wedge P_2 \sqsubseteq P_e$. Таким образом, для оценки корректности оптимизации нам необходимо зафиксировать эталонную модель вычисления.

Для введения количественных оценок эффективности для последующей оптимизации необходимо ввести внутренние переменные для оптимизации и задать соотношения, описывающие затраты для каждого перехода из состояния $\sigma$ в состояние $\sigma'$ и действия $\tau$ – $C(\tau, \sigma, \sigma')$. Соответственно, изменяется внутренняя спецификация

$$C_P = \bigvee_{\tau \in A, \sigma \in \overline{v}, \sigma' \in \overline{v}} C(\tau, \sigma, \sigma')$$

$$\Pi(P) = \Theta \vee \Box[N_P]_{\overline{v}} \vee \Box[R_P]_{\overline{v}} \vee \Box[C_P]_{\overline{v}}.$$

**Полная модель приложения**

Таким образом, расширяя описание приложения, мы получаем следующую формальную модель:

**Определение 10.** *Программа описывается следующими сущностями:*

1. *программа без времени $P(\overline{v}, \overline{x}, \Theta, A)$;*
2. *набор сбоев $F$ и соответствующая программа со сбоями с условием $F(P, F) \sqsubseteq P$;*
3. *счетчик часов now [1] со свойствами времени 1;*
4. *Внутренние состояния, описывающие наличие ресурсов, таких как доступная память;*
5. *функции расхода ресурсов $L_i(\tau)$ и $U_i(\tau)$, задающие границы потребления ресурсов для каждого действия из $P$;*
6. *Ценовые функции $C(\tau, \sigma, \sigma')$, используемые для оптимизации.*
7. *Внутренние и внешние спецификации $\Pi(P)$ и $\Phi(P)$.*

**Выводы**

В данной работе мы описали формальную модель процесса вычислений, позволяющую описывать и доказывать свойства устойчивых к сбоям вычислений. Эта модель может являться основой для применения формальных методов в среде контролируемого выполнения и обеспечивающей разработку и выполнение встраиваемых приложений [5].

**Литература**

1. Liu Z., Joseph M. Real-Time and Fault-Tolerant Systems. Specification, verification, refinement and scheduling. — UUNU/IIST, 2005.

2. Pnueli A. The temporal logic of programs // 18th Annual Symposium on Foundations of Computer Science. — 1977. — Pp. 46–57.

3. Lamport L. The temporal logic of actions // ACM Transactions on Programming Languages and Systems. — 1994. — Vol. 16(3). — Pp. 872–923.

4. Compositional quantitative reasoning / K. Chatterjee, L. de Alfaro, M. Faella et al. // ACM. — 2007.

5. Вьюкова Н. И., Галатенко В. А., Костюхин К. А., Шмырев Н. В. Организация отладочного комплекса для целевых систем со сложной архитектурой // Информационная безопасность. Микропроцессоры. Отладка сложных систем / под ред. Бетелина. — М.: НИИСИ РАН, 2004. — С. 120–150.

# ИССЛЕДОВАНИЕ ГРУППОВЫХ СВОЙСТВ УМНОЖЕНИЯ С ПАРАМЕТРОМ

**Годнева А.В. (МГУ им. М.В.Ломоносова)**
*god139@yandex.ru*

**Введение**

В республике Узбекистан при создании электронной подписи находит широкое применение умножение с параметром. Оно задается парой натуральных чисел (n,R) и определяется следующим образом на элементах $a, b \in \mathbb{Z}_n$.

$$a\circledR b = a + b + abR(mod\, n)$$

Заметим, что это коммутативная операция, и любое число, умноженное с параметром на 0, остается неизменным.

Для обоснования безопасности использования умножения с параметром в алгоритме электронной подписи [1] необходимо выяснить возможные порядки элементов относительно возведения в степень с параметром. В случае взаимной простоты $R$ и $n$ это было сделано в работе [2]. Ниже приводится описание групповой части алгебры $(\mathbb{Z}_n, \circledR)$.

**Основной результат**

**Теорема.** *Пусть* $n = \prod\limits_{k=1}^{q} p_k^{\alpha_k} * n_0$, $R = \prod\limits_{m=1}^{q'} p_m^{\beta_m} * r_0$, *причем* $R < n$, $(r_0, n) = 1$, $p_k, p_m$ — *простые множители в порядке возрастания, и для любого* $m, 1 \le m \le q'$, $p_m$ *делит* $n$. *Тогда группа обратимых элементов* $\mathbb{Z}_n^{\circledR}$ *алгебры* $(\mathbb{Z}_n, \circledR)$ *представима следующим образом:*

$$\mathbb{Z}_n^{\circledR} \cong \mathbb{Z}_{n_0}^* \times \mathbb{Z}_2 \times \mathbb{Z}_{2^{\alpha_1 - 1}} \times \mathbb{Z}_{p_2^{\alpha_2}} \times \ldots \times \mathbb{Z}_{p_q^{\alpha_q}} \text{ при } p_1 = 2 \text{ и } \beta_1 = 1;$$

$$\mathbb{Z}_n^{\circledR} \cong \mathbb{Z}_{n_0}^* \times \mathbb{Z}_{p_1^{\alpha_1}} \times \mathbb{Z}_{p_2^{\alpha_2}} \times \ldots \times \mathbb{Z}_{p_q^{\alpha_q}} \text{ в остальных случаях.}$$

**Вспомогательные утверждения**

Обозначим возведение числа $a$ в степень $t$ относительно операции $\circledR$ через $a^{\backslash t}$. Пусть $n = \prod\limits_k p_k^{\alpha_k}$, $R = \prod\limits_m p_m^{\beta_m}$, причем простые множители написаны в порядке возрастания. Обозначим $n_i = p_i^{\alpha_i}$, $n = \prod\limits_i n_i$. Если мы решим задачу возведения в степень отдельно для каждого $n_i$, то получим набор $a_i^{\backslash t}$, такой, что $a^{\backslash t} \equiv a_i^{\backslash t} (mod\, n_i) \forall i$.

Искомое число найдем, применяя, например, алгоритм Гарнера [3] для решения системы модулярных уравнений.

Если $(n_i; R) = 1$, то формула возведения в степень для обратимого $a$ получена в работе [2]:

$$a^{\backslash t} = \frac{(1 + R * a)^t - 1}{R} (mod\, n).$$

В противном случае, будем пользоваться леммой.

**Лемма 1** *Пусть $n = p^\alpha$, $R = p^\beta * r$, $\beta < \alpha$. Тогда формула для возведения в степень будет выглядеть следующим образом:*

$$a^{\backslash t} = ta + RC_t^2 a^2 + ... + R^{k-1} C_t^k a^k,$$

*где $k = \lceil \frac{\alpha}{\beta} \rceil$.*

Доказательство первой леммы несложно провести по индукции с использованием основных свойств биномиальных коэфициентов.

Теперь будем раскладывать группу в произведение циклических. Рассмотрим несколько случаев.

**Случай 1:** $(n, R) = 1$.

**Лемма 2.***В случае взаимнопростых $n$ и $R$ $\mathbb{Z}_n^{®} \simeq \mathbb{Z}_n^*$.*

*Доказательство.*

Этот факт доказывается непосредственным построением изоморфизма. Элементу $a \in \mathbb{Z}_n^{®}$ сопоставим элемент $\breve{a} \in \mathbb{Z}_n^*$ по формуле $\breve{a} = 1 + Ra$. То, что это отображение является изоморфизмом, проверяется непосредственно, с использованием критерия обратимости в алгебрах с параметром из работы [2] (напомним, что элемент в $\mathbb{Z}_n$ обратим относительно операции ® тогда и только тогда, когда $(1 + Ra)$ взаимно прост с $n$, то есть обратим в $\mathbb{Z}_n$ по умножению).

Группа $\mathbb{Z}_n i^*$ известным образом раскладывается в произведение циклических (см. [4]).

**Случай 2:** $n = p^\alpha$, $R = p^\beta * r$, $\beta < \alpha$, $p \neq 2$.

В этом случае воспользуемся следующей леммой.

**Лемма 3.** *Пусть $n = p^\alpha$, $R = p^\beta * r$, $\beta < \alpha$, $p \neq 2$. Тогда группа $\mathbb{Z}_n^{®}$ циклична и 1 является ее образующим элементом.*

Эта лемма доказывается с использованием факта, что порядок элемента делит порядок группы, и проверки, что порядок единицы не может отличаться от порядка группы.

**Случай 3:** $n = p^\alpha$, $R = p^\beta * r$, $\beta < \alpha$, $p = 2$.

Сформулируем еще одну лемму.

**Лемма 4.** *Пусть* $n = p^{\alpha}$, $R = p^{\beta} * r$, $\beta < \alpha$, $p = 2$. *Тогда* $\mathbb{Z}_n^{\circledR} \simeq \mathbb{Z}_2 \times \mathbb{Z}_{\frac{n}{2}}$ *или* $\mathbb{Z}_n^{\circledR} \simeq \mathbb{Z}_n$.

Данное утверждение доказывается аналогично предыдущему.

**Доказательство теоремы**

Рассмотрим элемент $a$, обратимый по умножению с параметром, и сопоставим ему кортеж $(a_0, a_1 ... a_q)$, где

$a_0 \equiv a \,(mod\, n_0)$

$a_1 \equiv a \,(mod\, n_1 = p_1^{\alpha_1})$

. . .

$a_q \equiv a \,(mod\, n_q = p_q^{\alpha_1})$. Покажем, что введенное отображение является изоморфизмом $\mathbb{Z}_n^{\circledR}$ и прямого произведения $\mathbb{Z}_{n_i}^{\circledR}$ по $i$ от 0 до $q$. Заметим, что все $a_i \in \mathbb{Z}_{n_i}^{\circledR}$. Действительно, это нужно доказывать только для $a_0$, так как в остальных $\mathbb{Z}_{n_i}^{\circledR}$ обратимы все элементы. Предположим, $a_0$ не является обратимым. Тогда $(1 + Ra_0, n_0) \neq 1$. Так как по построению $a = a_0 + n_0 x_0$ для некоторого целого неотрицательного $x_0$, $1 + Ra = (1 + Ra_0) + Rn_0 x_0$, и следовательно $(1 + Ra, n) \neq 1$, что противоречит предположению об обратимости $a$.

В силу взаимной простоты $n_i$ отображение является взаимно однозначным (см.[3]). Остается заметить, что оно выдерживает умножение с параметром. Возьмем обратимое b и разложим его аналогичным образом. Рассмотрим произведение с параметром a и b.

$a \circledR b = a + b + Rab = a_i + x_i n_i + b_i + y_i n_i + R(a_i + x_i n_i)(b_i + y_i n_i) \equiv a_i \circledR b_i \,(mod\, n_i)$.

Значит это искомый изоморфизм из $\mathbb{Z}_n^{\circledR}$ в произведение $\mathbb{Z}_{n_i}^{\circledR}$, а в каждом $\mathbb{Z}_{n_i}^{\circledR}$ разложение в прямое произведение было уже произведено в лемма 2, 3 и 4.

Автор выражает глубокую благодарность своему научному руководителю, н.с. А.В. Галатенко, за постановку задачи и внимание к работе.

**Литература**

1. O'z DSt 1092:2009. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи. — Узбекское агентство стандартизации, метрологии и сертификации, Ташкент, 2009.

2. Ишматова Ю. А. О некоторых свойствах групп алгебр с параметрами // Интеллектуальные системы, т.15, вып. 1–4, 2011.

3. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии — М.: МЦНМО, 2003.

4. Коблиц Н. Курс теории чисел и криптографии —Москва: Научное изд-во ТВП, 2001.

# ОБ ОДНОЗНАЧНОСТИ АЛФАВИТНОГО ДЕКОДИРОВАНИЯ

**Дергач П.С.**

Целью этой работы является установление алгоритмической разрешимости про-блемы однозначности алфавитного декодирования регулярных текстов.

**Введение**

А. А. Марковым было показано, что проблема однозначности алфавитного декоди-рования всех текстов над заданным конечным алфавитом $A$ сводится к декодирова-нию конечного числа слов над алфавитом $A$ длины не большей некоторой вычислимой величины, зависящей от длины схемы кодирования, мощности алфавита $A$ и др. параметров [1]. В предлагаемой работе исследован случай, когда кодируемое множес-тво слов является любым регулярным множеством. Показывается, что результат Маркова А. А. может быть обобщен на случай алфавитного декодирования любого регулярного множества слов над алфавитом $A$.

**Основные понятия и результаты.**

*Абстрактным конечным автоматом* называется набор $V = (A, Q, B, \varphi, \psi)$, где $A, Q, B$ - конечные множества,$\varphi$ - функция, определенная на множестве $Q \times A$ и принимающая значения из $Q$, $\psi$ - функция, определенная на множестве $Q \times A$ и принимающая значения из $B$. Множества $A, Q, B$ называются соответственно *входным алфавитом*, *алфавитом состояний* и *выходным алфавитом* автомата $V$. Функ-ция $\varphi$ называется *функцией переходов*, а функция $\psi$ - *функцией выходов* автомата $V$. *Входными словами* автомата $V$, $V = (A, Q, B, \varphi, \psi)$ называем произвольные конечные последовательности символов алфавита $A$. Для удобства рассматриваем при этом также "пустое"слово, не имеющее ни одного символа и обозначаемое $\Lambda$. *Выходными словами* алфавита $V$ называем конечные последовательности символов алфавита $B$, *словами состояний* - конечные последовательности символов алфавита $Q$ (в обоих случаях допускается и пустое слово $\Lambda$). Для каждого состояния автомата $V$ можно рассмотреть набор $(A, Q, B, \varphi, \psi, q)$, определяющий автомат $V$ с выделенным началь-ным состоянием $q$. Такие наборы $(A, Q, B, \varphi, \psi, q)$ называются *инициальными абс-трактными конечными автоматами*; для них используется обозначение $V_q$.

Введем ряд понятий, связанных со словами. Пусть $C$ - некоторое конечное множест-во. Если $\gamma = c(1)\dots c(n)$ - конечная последовательность символов $c(1),\dots,c(n)$ алфа-вита $C$, то говорим, что $\gamma$ есть *слово в алфавите* $C$. Число $n$ называем *длиной* слова $\gamma$ и обозначаем через $|\gamma|$. Длина пустого слова равна 0. Если $\gamma$ и $\delta$ - слова, причем $\gamma = \delta\delta'$ для некоторого слова $\delta'$, то говорим, что $\delta$ - *начало* слова $\gamma$, $\delta'$ - *конец* слова $\gamma$. Множество всех слов в алфавите $C$ обозначаем $C^*$. Начало слова $\gamma$, имеющее длину $l$, обозначаем $[_l(\gamma)$; окончание слова $\gamma$, имеющее длину $l$, обозначаем через $]_l(\gamma)$. Обозначим через $\gamma_{l,m} := ]_{m-l}([_m(\gamma))$, где $|\gamma| \geq m > l \geq 1$.

Функции переходов и выходов алфавита $V = (A, Q, B, \varphi, \psi)$ доопределим на мно-жестве $Q \times A^*$ (сохраним за ними те же обозначения). Именно, полагаем по определе-нию

$$\varphi(q,\Lambda) = q,\ \varphi(q,\alpha a) = \varphi(\varphi(q,\alpha),a),$$

где $q \in Q,\ \alpha \in A^*,\ a \in A$. Аналогично,

$$\psi(q,\Lambda) = \Lambda,\ \psi(q,\alpha a) = \psi(\varphi(q,\alpha),a).$$

Пусть $V_q = (A, Q, B, \varphi, \psi, q)$ - инициальный абстрактный конечный автомат,$B' \subseteq B$. Множество $M = \{\alpha \mid \alpha \in A^*,\ \psi(q,\alpha) \in B'\}$ называем *представимым в конечном автомате* $V_q$ с помощью подмножества $B'$ выходных символов. Говорим также, что автомат $V_q$ *представляет* $M$ *посредством* $B'$. Пусть $M \subseteq A^* \setminus \{\Lambda\}$. Если существует конечный автомат $V_q$, представляющий событие $M$ посредством некоторого подмно-жества $B' \subseteq B$, то событие $M$ называем *представимым*.

Введем понятие *обобщенного источника*. *Обобщенным источником* в алфавите $A$ назовем конечный ориентированный граф $G$, у которого выделены начальная и финальная вершины $v, w, v \neq w$, причем каждому ребру приписано пустое слово $\Lambda$ либо символ алфавита $A$. Допускается наличие в графе петель и кратных ребер. *Путем* в обобщенном источнике $G$ будем называть последовательность $\pi = (v_1, \rho_1, v_2, \rho_2, \dots, \rho_n, v_{n+1})$, где $v_1, v_2, \dots, v_{n+1}$ - вершины графа $G$, $\rho_i$ - ребро графа $G$, ведущее от вершины $v_i$ к вершине $v_{i+1}$, $i = 1,\dots,n,\ n \geq 1$. Пути $\pi$ сопоставляем слово $[\pi] = a_1 \dots a_n$, где $a_i$ - символ алфавита $A$ либо пустое слово $\Lambda$, приписанное ребру $\rho_i,\ i = 1,\dots,n$. Говорим, что путь $\pi$ *ведет от вершины* $v_1$ *к*

*вершине* $v_{n+1}$. Пусть $\alpha \in A^* \setminus \{\Lambda\}$, $u$ - вершина обобщенного источника $G$, множество всех вершин $u'$ обобщенного источника $G$, для которых существует путь $\pi$, ведущий от $u$ к $u'$ и такой, что $[\pi] = \alpha$, обозначим $\theta(u, \alpha)$. Каждый обобщенный источник $G$ с начальной вершиной $v$ и финальной вершиной $w$ определяет событие $|G| = \{\alpha \mid \alpha \in A^* \setminus \{\Lambda\}, w \in \theta(v, \alpha)\}$.

Пусть $A = \{a_1, \ldots, a_r\}$ - произвольный конечный непустой алфавит. Пусть $P_1, P_2$ - непустые множества слов в алфавите $A$. Здесь и далее для удобства пустое слово за элемент множества $A^*$ не считается. Определим следующие операции над $P_1$ и $P_2$:

1. *Произведение* множеств $P_1$ и $P_2$ (обозначаем $P_1 \cdot P_2$) есть множество всех слов вида $\alpha_1\alpha_2$, где $\alpha_1 \in P_1,\ \alpha_2 \in P_2$.

2. *Итерация* множества $P_1$ (обозначаем $(P_1)^*$) есть множество всех слов вида $\alpha_1 \ldots \alpha_k$, где $\alpha_1 \in P_1,\ \ldots, \alpha_k \in P_1,\ k \geq 1$.

Введем понятие регулярного множества в алфавите $A$. Множество $P$, $P \subseteq A^*$, называем *регулярным в алфавите* $A$, если его можно получить из множеств вида $\{a\}$, $a \in A$, применением конечного числа операций $\cup, \cdot, ()^*$. Более подробно, опреде-ление регулярных множеств таково:

1. $\{a\}$, где $a$ - произвольная буква алфавита $A$, - регулярное множество в алфавите $A$;

2. Если $P_1, P_2$ - регулярные множества в алфавите $A$, то $P_1 \cup P_2,\ P_1 \cdot P_2,\ (P_1)^*$ - регулярные множества в алфавите $A$;

3. Регулярность произвольного множества в алфавите $A$ устанавливается в соответ-ствиями с пп. 1, 2 за конечное число шагов. Введем понятие регулярного выражения в алфавите $A$. Регулярное выражение в алфавите $A$ представляет собой слово в алфавите $A \cup \{\vee, \cdot, (, ), *\}$, определяемое следующим образом:

1. Буквы алфавита $A$ - регулярные выражения в алфавите $A$;

2. Если $\alpha,\ \beta$ - регулярные выражения в алфавите $A$, то $(\alpha \vee \beta),\ (\alpha \cdot \beta),\ (\alpha)^*$ - регулярные выражения в алфавите $A$;

3. Регулярность произвольного выражения в алфавите $A$ устанавливается в соот-ветствиями с пп. 1, 2 за конечное число шагов.

Сопоставим индуктивно каждому регулярному выражению $\mathfrak{P}$ в алфавите $A$ регу-лярное множество $|\mathfrak{P}|$ в алфавите $A$:

1. Множество $\{a\}$ - в случае $\mathfrak{P} = a,\ a \in A$;

2. Множество $|\mathfrak{P}_1| \cup |\mathfrak{P}_2|$ - в случае $\mathfrak{P} = (\mathfrak{P}_1 \vee \mathfrak{P}_2)$;

3. Множество $|\mathfrak{P}_1| \cdot |\mathfrak{P}_2|$ - в случае $\mathfrak{P} = (\mathfrak{P}_1 \cdot \mathfrak{P}_2)$;

4. Множество $(|\mathfrak{P}_1|)^*$ - в случае $\mathfrak{P} = (\mathfrak{P}_1)^*$.

**6.** Зафиксируем два конечных непустых алфавита $A$ и $B$. Пусть есть какое-то отображение $f : A \to B^*$:

$$f(a_1) = \beta_1$$

$$f(a_2) = \beta_2$$

$$\dots$$

$$f(a_r) = \beta_r$$

Это соотношение называется *схемой кодирования*. Доопределим отображение $f$ до отображения $\tilde{f} : A^* \to B^*$ следующим образом:

$$\tilde{f}(a_{i_1} a_{i_2} \dots a_{i_n}) = \beta_{i_1} \beta_{i_2} \dots \beta_{i_n}.$$

Это отображение $\tilde{f}$ будем называть *алфавитным кодированием*.

Пусть есть некоторое регулярное множество $P$ в алфавите $A$ и некоторое алфавит-ное кодирование $f$. Пусть $\beta \in \tilde{f}(P)$. Тогда $\alpha \in P$ называется *расшифровкой $\beta$ при алфавитном кодировании $\tilde{f}$ на регулярном множестве $P$* или *расшифровкой $\beta$*, если $f(\alpha) = \beta$. Таких расшифровок может быть несколько. Если для любых различных $\alpha_1, \alpha_2 \in P$ выполняется $\tilde{f}(\alpha_1) \neq \tilde{f}(\alpha_2)$, то говорим, что *декодирование однозначно на $P$ по $\tilde{f}$*.

**Теорема 1.** *Существует алгоритм проверки однозначности алфавитного декоди-рования для любого регулярного текста.*

**Литература**

1. С. В. Яблонский. Введение в дискретную математику.

2. В. Б. Кудрявцев, С. В. Алешин, А. С. Подколзин. Введение в теорию автоматов.

# МАТЕМАТИЧЕСКОЕ ОЖИДАНИЕ СРЕДНЕЙ ДЛИНЫ КОДОВ ХАФМАНА

**Кучеренко Н.С. (Московский государственный университет имени М.В. Ломоносова)**
*nsk.email@gmail.com*

Коды Хафмана и алгоритмы их построения [1] широко применяются при сжатии информации. При построении кода Хафмана полагается, что кодируется алфавит с известными вероятностями появления его символов. На практике значения этих вероятностей могут быть не известны, и для их оценивания необходимо провести предварительную работу, например — вычислить частоту появления символа в сжимаемом файле. Частоту появления символа назовем *весом* символа. В силу того, что предварительное вычисление весов может быть трудоемко или невозможно (сжимаемый файл сразу весь не доступен), теоретический интерес представляет следующая задача. Предположим, что веса символов являются случайными величинами, каково тогда математическое ожидание средней длины кода Хафмана.

Обозначим вес символов кодируемого алфавита с мощностью $n$ через $(w_1, \ldots, w_n)$ и положим, что это случайные величины, которые задаются следующим образом. Рассмотрим $x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n-1)}$ — вариационный ряд построенный по реализации выборки из закона с функцией распределения $F$, определенной на интервале $(a, b)$. Величину $w_i$ зададим как

$$w_i = x_{(i)} - x_{(i-1)},\ i = 1, \ldots, n,\ x_{(0)} = a, x_{(n)} = b.$$

Обозначим через $T_n^m(F)$ математическое ожидание средней длины кода Хафмана для кодируемого алфавита мощности $n$, кодирующего алфавита мощности $m$ и весов, задаваемых с помощью функции распределения $F$. В работе исследуется поведение функции $T_n^m(F)$ при увеличении мощности кодируемого алфавита $n$ и фиксированных параметрах $m$ и $F$.

Средняя длина кода Хафмана оценивается с помощью энтропии с точностью до константы. Для весов $(w_1, \ldots, w_n)$ и мощности кодирующего алфавита $m$ энтропия $H_m(w_1, \ldots, w_n)$ определяется формулой $-\sum_{i=1}^{n} w_i \cdot \log_m w_i$. При изучении поведения математического ожидания средней длины кода Хафмана можно перейти к изучению

поведения математического ожидания энтропии $H_m(w_1, \ldots, w_n)$ как функции от случайных весов. В работах автора [2–5] применяется похожая техника, поэтому их результаты можно перенести для случая кода Хафмана.

Автором получены достаточные условия на функцию распределения $F$, при которых математическое ожидание средней длины кода Хафмана имеет порядок логарифма (теорема 1), и уточнение этих условий до получения асимптотики (теорема 2).

**Теорема 1.** *Пусть существует невырожденный интервал* $(c, d)$, $(c, d) \subseteq (a, b)$, *и положительные вещественные константы* $c_1$ *и* $c_2$, *для которых выполнено*

$$\forall x, y \in (c, d),\ x > y, \quad c_1 \cdot (x - y) \geq F(x) - F(y) \geq c_2 \cdot (x - y).$$

*Тогда*

$$T_n^m(F) \asymp \log_m n \quad (n \to \infty).$$

Для формулировки теоремы 2 понадобятся следующие определения. Обозначим носитель функции $f$, определенной на интервале $(a, b)$, через $supp(f)$, $supp(f) = \{x \in (a, b) : f(x) \neq 0\}$. Назовем функцию $f$ *функцией с конечно-интервальным носителем,* если $supp(f)$ имеет вид $supp(f) = \sqcup_{i=1}^{s}(a_i, b_i) \sqcup K$, где $a_i < b_i$, $b_i < a_{i+1}$, $\forall i = 1 \ldots (s-1)$, $a_s < b_s$, и множество точек $K$ имеет меру ноль.

**Теорема 2.** *Пусть функция распределения* $F$ *представима в виде суммы двух функций — функции скачков* $F'$ *и абсолютно непрерывной функции* $F''$. *Пусть производная* $f$ *функции* $F''$ *имеет конечно-интервальный носитель, ограничена и отделена от нуля на множестве* $supp(f)$ *и интегрируем по Риману, а функция* $F'$ *имеет конечное число скачков. Тогда*

$$T_n^m(F) \sim \log_m n \cdot \int_B 1\, dx \quad (n \to \infty),$$

*где* $B = supp(f)$.

В следующей теореме изучено поведение математического ожидание средней длины кода Хафмана как ограниченной функции от мощности кодируемого алфавита $n$.

**Теорема 3.** *Существуют такая функция распределения* $F$, *что*

$$\forall n \in \mathbb{N} \qquad T_n^m(F) = 1.$$

*Для любого вещественного $b > 1$ существуют такая функция распределения $F$, что*

$$b + 2 \gtrsim T_n^m(F) \gtrsim b \quad (n \to \infty).$$

Также автором исследован случай когда математическое ожидание средней длины кода Хафмана $T_n^m(F)$ является возрастающей функцией по порядку меньшей, чем логарифм. В теореме 4 описано семейство $S$ возможных асимптотик функции $T_n^m(F)$, а в теореме 5 семейство $S^*$ возможных порядков.

Для описания семейства $S$ понадобится следующее определение. Положительная, возрастающая функция $r(x)$ называется *сохраняющей асимптотику,* если выполнено условие

$$\forall c \in \mathbb{R} \quad r(x + c) \sim r(x) \quad (x \to \infty).$$

Семейство $S$ возможных асимптотик промежуточных функций роста состоит из функций вида $r(\log_m \log_m(n))$, где возрастающая, положительная и дифференцируемая функция $r(x)$, определенная на интервале $(x_0, +\infty), x_0 \geq 0$, сохраняет асимптотику и имеет в качестве производной монотонную, положительную и непрерывную функцию $r'(x)$, удовлетворяющую условию:

$$\exists \alpha > 0, \alpha \in \mathbb{R} : \overline{\lim_{x \to +\infty}} \frac{r'(x)}{x^\alpha} < 1.$$

Все функции из $S$ являются возрастающими и имеют порядок меньше чем $\log_m n$ при $n \to \infty$.

**Теорема 4.** *Для любой функции $r(\log_m \log_m(n))$ из семейства $S$ существуют функция распределения $F$ такая, что*

$$T_n^m(F) \sim r(\log_m \log_m(n)) \quad (n \to \infty).$$

Для формулировки следующей теоремы понадобится понятие сохраняющей порядок функции. Положительная, возрастающая функция $r(x)$ называется *сохраняющей порядок,* если выполняется условие

$$\forall c \in \mathbb{R}, c \neq 0, \quad r(c \cdot x) \asymp r(x) \quad (x \to \infty).$$

Семейство $S^*$ возможных порядков промежуточных функций роста состоит из функций вида $r(\log_m(n))$, где неограниченно возрастающая, положительная и дифференцируемая функция $r(x)$, определенная на интервале $(x_0, +\infty)$, $x_0 \geq 0$, сохраняет порядок и имеет в качестве производной монотонную, положительную и непрерывную функцию $r'(x)$, удовлетворяющую условию:

$$\exists \alpha \in \mathbb{R}, 0 < \alpha < 1 : \overline{\lim_{x \to +\infty}} \frac{r'(x)}{x^{\alpha-1}} \leq 1.$$

Все функции из $S^*$ являются возрастающими и имеют порядок меньше чем $\log_m n$ при $n \to \infty$. В отличие от класса $S$, в классе $S^*$ есть функции по порядку большие чем любая функция из $S$, например $(\log_m n)^\alpha$, $0 < \alpha < 1$. Автором показано, что для функций семейства $S^*$ верна следующая теорема

**Теорема 5.** *Для любой функции $r(\log_m(n))$ из семейства $S^*$ существуют функция распределения $F$ такая, что*

$$T_n^m(F) \asymp r(\log_m(n)) \quad (n \to \infty).$$

## Литература


1. Яблонский С. В. Введение в дискретную математику. — М.: Высшая школа, 2002.

2. Кучеренко Н. С. Сложность поиска идентичных объектов в случайных базах данных // Интеллектуальные системы. — М.: Изд-во РГГУ, 2007, т. 11, № 1–4, с. 525–550.

3. Кучеренко Н. С. О промежуточных функциях роста сложности поиска для случайных баз данных // Интеллектуальные системы. — М.: Изд-во РГГУ, 2009, т. 13, № 1–4, с. 361–395.

4. Кучеренко Н. С. Задача поиска по ключу с определением позиции // Интеллектуальные системы. — М.: Изд-во РГГУ, 2010, т. 14, № 1–4, с. 293–306.

5. Кучеренко Н. С. Средняя сложность поиска идентичных объектов для случайных неравномерных баз данных // Дискретная математика. — М.: 2011, т. 23, № 2, с. 129–158.

# ОБ ОДНОМ ПОДХОДЕ К ЛИНЕЙНОЙ АДАПТИВНОЙ ЦИФРОВОЙ ОБРАБОТКЕ СИГНАЛОВ

**Мазуренко И.Л. (мех-мат факультет МГУ им. М.В. Ломоносова, кафедра МаТИС)**
*ivan@mazurenko.ru*

Адаптивная линейная система с обратной связью представляет собой адаптивный алгоритм, получающий на вход разность результата обработки входного сигнала $x$ некоторым устройством обработки информации и требуемого выходного сигнала $d$ ($x-d=\varepsilon$ - сигнал ошибки), и итерационно подстраивающий параметры устройства обработки информации с целью минимизации сигнала ошибки [1].

Примерами адаптивных систем являются системы линейного предсказания сигнала, широко применяющаеся при цифровом кодировании речи, системы моделирования и идентификации, применяющиеся для изучения вибраций механических системы, системы выравнивания характеристик, использующиеся для исключения влияния каналов связи, системы подавления шумов и помех, применяющиеся в задачах адаптивной шумоочистки и адаптивного эхоподавления. Рассмотрению алгоритмических подходов к решению последней задачи и посвящена настоящая работа.

В задаче адаптивной линейной фильтрации предполагается, что устройство обработки входной информации - линейно, и адаптивный алгоритм используется для итеративного подстраивания параметров этой линейной системы, а именно коэффициентов $H = (h_0, \dots h_{L-1})$ линейного фильтра с конечной импульсной характеристикой. При этом выходной сигнал $y$ в момент времени $k$ получается путем свертки входного сигнала $x$ и коэффициентов фильтра: $y_k = \sum_{l=0}^{L-1} h_l x_{k-l}$. Наиболее распространенным применением адаптивной линейной фильтрации на практике является задача подавления эхо-сигнала в телефонных (проводных и беспроводных) сетях. Эхо в таких сетях делится на электрическое (возникающее в т.н. «гибридах» - устройствах преобразования 2-проводных телефонных сетей в 4-проводные) и акустическое (возникающее в оконечном оборудовании: мобильных и стационарных телефонах, устройствах обратной связи, оконечных устройствах IP-телефонии).

Адаптивные алгоритмы подстройки параметров линейного фильтра основываются на принципе минимизации квадрата сигнала

ошибки $\varepsilon$. В качестве параметра сигнала ошибки рассматривают оценку величины среднеквадратического отклонения сигнала $E(\varepsilon_k^2) = E(d_k^2) + H^T E(X_k X_k^T) H - 2E(d_k X_k^T) H$.

Поскольку величина среднеквадратического отклонения сигнала ошибки представляет собой положительно определенную квадратичную форму от коэффициентов фильтра $H$, минимум квадрата ошибки достигается в точке равенства нулю градиента $\nabla(\varepsilon_k^2) = 2RH - 2P$, где $R = E(X_k X_k^T)$ – LxL-матрица автокорреляции, $P = E(d_k X_k^T)$ – корреляция эхо и требуемого выходного сигнала $d$. Это дает нам известную теорему Винера-Хопфа [1], позволяющую найти «идеальный» фильтр H, минимизирующий квадрат ошибки предсказания: $H = R^{-1} P$.

Поскольку на практике ни оценка матрицы автокорреляции $R$, ни вектор $P$ точно неизвестны, используют итерационные методы, в той или иной степени сводящиеся к применению метода градиентного спуска: $H_{k+1} = H_k + \mu(-\nabla_k)$, $0 < \mu < \frac{1}{\lambda_{max}}$, где $\lambda_{max}$ - максимальное по модулю собственное значение матрицы автокорреляции $R$.

Наиболее простой и распространенной модификацией метода градиентного спуска является т.н. «метод наименьших квадратов», который применительно к данной задаче сводится к тому, что в качестве оценки математического ожидания квадрата ошибки $E(\varepsilon_k^2)$ берется величина $\varepsilon_k^2$. Формула адаптивного обновления оценки коэффициентов фильтра получается значительно более простой вычислительно: $H_{k+1} = H_k - \mu \hat{\nabla}_k = H_k + 2\mu\varepsilon_k X_k$, $0 < \mu < tr(R)$, ибо требует $2L + 2$ умножений. Более быстро сходящийся алгоритм нормализованных наименьших квадратов использует $2L + 2$ умножений и одно деление: $H_{k+1} = H_k + 2\mu\varepsilon_k \frac{X_k}{\|X_k\|^2}$, $0 < \mu < 1$.

В конце XX - начале XXI века было предложено несколько алгоритмов цифровой обработки сигналов, более эффективных с точки зрения соотношения скорости сходимости и вычислительной сложности. К их числу можно отнести метод аффинных проекций ([2]), метод быстрых аффинных проекций ([3,4]) и приближенные методы быстрых аффинных проекций, основанные на Теплицевом приближении оценки автокорелляционной матрицы $R$ ([5] и др.).

Метод аффинных проекций, обладающий наивысшей из известных методов скоростью сходимости, имеет сложность $2LN + K_{inv} N^2$ умножений, где $K_{inv}$ — сложность обращения матрицы, а потому практически неприменим на практике. Его упрощение — метод быст-

рых аффинных проекций — обладает уже линейной сложностью относительно размера проекции $N$ и числа коэффициентов фильтра $L$ — $2L+20N$ умножений, – однако, не лишен недостатков, один из которых (неустранимое накопление ошибок при целочесленной реализации) делает его практически неприменимым. Приближенные методы алгоритма быстрых аффинных проекций, описанные [5], лишены данного недостатка и дают сложность в лучшем случае $2L+8N-3$ умножений, однако не обладают стабильностью в случае быстроменяющегося по своим спектральным характеристикам входного сигнала $x$.

Автором данной работы была предложена модификация приближенного метода быстрых аффинных проекций, основанного на применении Теплицевого приближения автокорреляционной матрицы $R$, в котором, в отличие от алгоритма [5], на каждом шаге итерации алгоритма Левинсона-Дурбина испольнуется новое приближение автокорреляционной матрицы, а для контроля за сходимостью алгоритма применяется пороговое правило контроля уровня недекоррелированной ошибки адаптации. Данная модификация алгоритма, практически не проигрывая в скорости методу [5], дает на тестовых данных значительно более высокую скорость сходимости и надежность работы алгоритма адаптации. Работа защищена патентом США.

**Литература**

1. Уидроу Б., Стирнз С. Адаптивная обработка сигналов. // М. — Радио и связь, 1989. — 440 с.

2. Ozeki K., Umeda T. An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties // Proc. of the Elec. Comm. Japan, vol. J67-A, February 1984, — стр. 126–132.

3. Gay S. L. A fast converging, low complexity adaptive filtering algorithm. // Third International Workshop on Acoustic Echo Control, Plestin les Greves, France, 7-8 Sept. 1993.

4. Tanaka M., KanedaY., Makino. S., Kojima J. A fast projection algorithm for adaptive filtering. // IEICE Trans. Fund. E78-A (10m), October 1995.

5. Ding H. Fast affine projection adaptation algorithms featuring stable symmetric positive-definite linear system solvers. // Applications of Signal Processing to Audio and Acoustics, 2005.

# РЕШЕНИЕ ЗАДАЧИ ОПТИМИЗАЦИИ БЕЗОПАСНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ

**Марков А.С. (Москва)**
*fadin.andrey@gmail.com*
**Патраков Н.В. (Москва)**
*nikolay.patrakov@gmail.com*
**Фадин А.А. (Москва)**
*fadin.andrey@gmail.com*

## Введение

Построение безопасных информационных систем является необходимым условием для функционирования всех современных государственных, общественных и коммерческих организаций.

Что следует понимать под безопасной информационной системой? В соответствии с Национальным стандартом Российской Федерации <Информационная технология. Практические правила управления информационной безопасностью> (ГОСТ Р ИСО/МЭК 17799-2005), информационная безопасность - защита конфиденциальности, целостности и доступности информации. Здесь, конфиденциальность - это свойство информационных ресурсов, в том числе информации, связанное с тем, что они не станут доступными и не будут раскрыты для неуполномоченных лиц. Целостность - это неизменность информации в процессе ее передачи или хранения. Доступность - это свойство информационных ресурсов, в том числе информации, определяющее возможность их получения и использования по требованию уполномоченных лиц.

Фактически обеспечением безопасности для информационной системы является выполнение приведенных выше трех свойств.

Что является оптимизацией системы? Изменение состава или структуры частей (компонентов) системы для достижения оптимального значения для одного или нескольких свойств системы с сохранением допустимых значений для всех остальных свойств.

В данной работе поставлена задача по построению компонентной модели информационной системы, которая:

1. подходит для оценки свойств системы, связанных с безопасностью;

2. удобна для выполнения преобразований с целью ее дальнейшей оптимизации.

Система - это множество элементов, находящихся в связях и отношениях друг с другом.

Для информационной системы предлагается использовать следующий подход.

Вводятся две различные сущности: компонент и интерфейс.

Соответственно безопасность системы зависит от выполнения:

1. требований по доверию (assuarance) ко всем ее компонентам и интерфейсам;

2. требований по функционалу ко всем ее компонентам и интерфейсам.

Математической моделью данной информационной системы будет являться ориентированный граф, вершины которого представляют собой компоненты безопасной информационной системы, а рёбра - интерфейсы и каналы коммуникации между ними.

Компоненты могут быть:

1. высокоуровневые сущности: СУБД, операционная система, веб-сервер, коммутационное устройство сети, ЭВМ;

2. сущности более низкого уровня: библиотеки, исполняемые модули, брокеры запросов удаленного вызова процедур;

3. сущности связанные с внешними системами и ролями пользователей: администратор, злоумышленник, клиентская машина. Как правило, в рамках моделирования их внутренний состав не рассматривается.

Для снижения числа связей и упрощения дальнейшего анализа принимается решение разделить все компоненты на два вида:

1. функциональные - связанные непосредственно с назначением системы;

2. инфраструктурные - обеспечивающие среду выполнения и установление коммутации между всеми остальными компонентами.

Безусловно, эта граница не всегда является четкой, некоторые компоненты могут сочетать в себе свойства двух типов, но инфраструктурные компоненты (гипервизор, драйвер файловой системы, брокер запросов CORBA) характеризует в целом большое количество однотипных связей, если же рассматривать их функционирование, то, как правило, они обрабатывают данные других компонентов и редко бывают инициаторами запросов.

Фактически, если рассматривать систему, любые два функциональных компонента взаимодействуют через инфраструктурный компонент, соответственно свойства (и требования по безопасности) для всех рёбер между функциональными компонентами можно сопоставить с необходимыми свойствами инфраструктурных элементов.

Очевидно, что в дальнейшем для оценки свойств системы достаточно исследования свойств вершин соответствующего ей графа.

Каждый элемент может быть связан с другим по двум основным требованиям: по чтению/выполнению и по записи. Кроме того, вводится метрика критичности связи, т.е. степень важности этой связи для зависимого компонента.

В первом случае возможно нарушение доступности в зависимом компоненте по результатам операций во влияющем компоненте. Во втором случае возможно нарушение целостности, доступности и конфиденциальности.

Дальнейший анализ системы целесообразно выполнять на основе матрицы, построенной по описанному выше графу.

Перечень возможных свойств объектов системы:

1. наименование компонента;
2. автор компонента;
3. срок существования (сколько времени пришло с начало первого внедрения и его использования);
4. количество найденных уязвимостей в компоненте за всё время его использования;
5. перечень типов данных ограниченного пользования, которые обрабатывает компонент;
6. требования по доступности компонента.

**Литература**

1. Марков А. С., Цирлов В. Л. Направления совершенствования методов сертификации программного обеспечения на отсутствие уязвимостей // Ежегодное совещание "Взаимодействие участников рынка продуктов и услуг в области безопасности. – М., 2007.

2. Anup K.Ghosh, Gary McGraw An Approach for Certifying Security in Software Components.: Reliable Software Technologies report, 2001.

# АТАКИ НА ПРОТОКОЛ НИДХЕМА-ШРЕДЕРА. МОДИФИКАЦИЯ ПРОТОКОЛА НИДХЕМА-ШРЕДЕРА И ПОСТРОЕНИЕ НА ЕГО ОСНОВЕ

**Мозгалева О.А. (Московский Государственный Университет имени М.В. Ломоносова)**
*omozgaleva@gmail.ru*

В настоящее время одним из важнейших сервисов информационной безопастности является аутентификация. Весьма известным решением проблемы аутентификации является протокол Нидхема-Шредера (R.M. Needham, M.D. Schroeder)[1], предложенный в Нидхемом и Шредером в 1978 году. Этот протокол использует для аутентификации третью доверенную сторону, формирующую сеансовый ключ для идентификации одной стороны перед другой.

Протокол аутентификации Нидхема-Шредера выглядит следующим образом.

1. Абонент $A$ вырабатывает случайное $r_A$ и отправляет центру $S$ запрос $m_A = (idA, idB, r_A)$.

2. Центр S вырабатывает сеансовый ключ $K$ и отправляет абоненту $A$ сообщение $\omega_S = E_{K_AS}(r_A, idB, K, E_{K_BS}(K, idA))$, зашифрованное ключом $K_AS$, разделенным между центром $S$ и абонентом $A$.

3. Абонент $A$ расшифровывает сообщение $\omega_S$ и проводит аутентификацию центра $S$ по параметру $r_A$. В случае успешной аутентификации абонент $A$ пересылает абоненту $B$ сообщение $y_B = E_{K_BS}(K, idA)$.

4. Абонент $B$ расшифровывает сообщение $y_B$, на основе знания ключа $K_BS$, разделенного им с центром $S$, извлекая при этом сеансовый ключ $K$. Абонент $B$ вырабатывает случайный параметр $r_B$, и отправляет сообщение $\omega_B = E_K(r_B)$ абоненту $A$.

5. Абонент $A$ расшифровывает сообщение $\omega_B$ на ключе $K$, определяет $r_B$, формирует и отправляет абоненту $B$ сообщение $\omega_A = E_K(\psi(r_B))$, где $\psi$ - некоторая заранее выбранная числовая функция.

6. Абонент $B$ расшифровывает сообщение $\omega_A$, вычисляет свое $\psi(r_B)$ и сравнивает результаты, тем самым аутентифицируя абонента A.

Протокол Нидхема-Шредера является самым известным протоколом аутентификации, однако он уязвим для атаки, изобретенной

в 1981 году Деннингом (Denning) и Сакко (Sacco)[3]. Эта атака основана на атаке с повторной передачей сообщения. Предполагается, что произошла компроментация сеансового ключа. Повторно отправив соответствующее этому ключу сообщение, злоумышленник может аутентифицироваться под видом законного абонента.

Мною был изучен протокол Нидхема-Шредера и получены следующие результаты.

Первый результат связан с ограничениями на функцию шифрования. При определенных свойствах функции шифрования $E$ один абонент сможет аутентифицироваться у другого абонента под видом некоторого третьего законного абонента.

**Теорема 1.** *Если в протоколе Нидхема-Шредера*

*1. Функция шифрования является гомоморфизмом относительно пар (K,id): $E_{K_BS}(K_1 \bullet K_2, idA_1 \circ idA_2) = E_{K_BS}(K_1, idA_1) * E_{K_BS}(K_2, idA_2)$, где " $\bullet$ " $\circ$ " $*$ " - групповые операции на соответствующих множествах.*

*2. Абоненту A известно разложение идентификатора законного абонента C относительно групповой операции: $idC = idA^{\alpha} \circ {idA_1}^{\alpha_1} \circ \ldots {idA_n}^{\alpha_n}$, где $A_i$ - законные участники. То абонент A с помощью абонентов $A_1, \ldots, A_n$ сможет аутентифицироваться у абонента B под видом законного абонента C.*

Второй результат показывает необходимость выбора параметра $r_A$ случайным для предотвращения возможности атаки, совмещающей в себе атаку "противник в середине"и атаку с повторной передачей сообщения. Данный результат очень важен, так как открывает большие возможности взлома протокола при успешных атаках на генератор случайных чисел.

**Теорема 2.** *Если параметр $r_A$ фиксирован, и происходит компроментация сеансового ключа K (использованного ранее в других сеансах или текущего), то становится возможной атака со стороны злоумышленника C совмещающей в себе атаку "противник в середине"и атаку с повторной передачей сообщения. В результате такой атаки:*

*1. Злоумышленник C аутентифицируется у абонента B под видом абонента A.*

*2. Более того, злоумышленник C сможет "подслушивать"все дальнейшие сообщения, передаваемые абонентом A абоненту B, и*

*наоборот.*

Первое утверждение теоремы очевидным образом следует в результате атаки Деннинга-Сакко. Второе утверждение получается в результате следующей атаки. На шаге 2 злоумышленник $C$ перехватывает новое сообщение $\omega'_S$ от центра аутентификации $S$ к абоненту $A$, и подменяет его старым сообщением $\omega_S = E_{K_AS}(r_A, idB, K, E_{K_BS}(K, idA))$, соответствующим старому ключу $K$. Так как абонент $A$, аутентифицирует центр $S$, расшифровывая сообщение $\omega_S$, по параметру $r_A$, который в свою очередь постоянен по условию теоремы, то аутентификация центра $S$ пройдет успешно и участник $A$ продолжит выполнение протокола. Таким образом, все последующие сообщения информационного взаимодействия абонентов $A$ и $B$, будет происходить на основе шифрования ключом $K$, который в свою очередь известен злоумышленнику $C$, а, значит, он сможет "подслушивать"все сообщения абонентов $A$ и $B$, которые даже не будут это подозревать.

Следующий результат представляет собой модификацию протокола Нидхема-Шредера. Полученная модификация, во-первых, позволяет предотвращать атаку с повторной передачей сообщения. Это реализовывается введением временной метки T. Во-вторых, она позволяет строить другие блочные протоколы, использующие несколько дополнительных серверов. Для этого вводится несколько заглушек, для последующего связывания нескольких модифицированных протоколов в один блочный протокол.

Модифицированная схема Нидхема-Шредера:

$$N - Sch - mode(r, idA_1, idS, idA_2, K_{A_1S}, \\ K_{A_2S}, T, id1, id2, T', K', K'', r') = (K)$$

Входные данные: $r$ - случайное число, $idA_1, idS, idA_2$ - идентификаторы абонентов и сервера аутентификации, $K_{A_1S}, K_{A_2S} K_{A_1S}, K_{A_2S}$ - долговременные ключи, распределенные между сервером аутентификации и абонентами, $T$ - время действия нового сеансового ключа. Заглушки: $id1, id2$ - идентификаторы законных пользователей, $T'$ - временная метка, $K', K''$ - ключи, $r'$ - случайное число. Выходные данные: $K$ - сеансовый ключ.

1. Абонент $A_1$ вырабатывает случайное $r$ и отправляет серверу $S$ сообщение $m_s = (r, idA_1, idA_2, id1, id2, T', K')$.

2. Сервер $S$ отправляет абоненту $A_1$ сообщения $x_S = E_{K_{A_1S}}(r, idA_2, id1, K, T)$, $x_{A_2} = E_{K_{A_2S}}(idA_1, idA_2, id1, K, T)$.

3. Абонент $A_1$ вырабатывает случайное $r'$ и отправляет абоненту $A_2$ сообщение $m_{A_2} = (r', Aut = E_K(idA_1, T, K''), x_{A_2})$.

Время $T$ - это время действия сеансового ключа $K$. Соответственно теперь противник сможет посылать повторно сообщение лишь в период времени $T$. Соответственно $T$ должно быть выбрано таким, чтобы за это время компроментация ключа $K$ была возможна с очень маленькой вероятностью.

Важным следствием этой модифицированной схемы является ее связь с протоколом Kerberos, который является самым актуальным и распространенны протоколом сетевой аутентификации. Протокол Kerberos представляется прямой суммой двух модифицированных схем Нидхема-Шредера с завершающей посылкой, для взаимной аутентификации.Более того в результате последовательного выполнения нескольких модфицированных схем Нидхема-Шредера можно получить протокол, использующий несколько вспомогательного сервера. Этот протокол в сравнении с модифицированным протоколом Нидхема-Шредера (однократным его применением) имеет важные преимущества. Во-первых, в очень больших сетях данный протокол позволяет уменьшить нагрузку на сервера, распределяя ее среди серверов, обслуживающих маленькие подсети. Во-вторых, позволяет ускорить процесс повторной аутентификации.

**Литература**

1. Гашков С.Б., Применко Э.А., Черепнев М.А, Криптографические методы защиты информации (1-е изд.) учеб. пособие, Издательство: ИЦ Академия, 2010г., Серия: Высшее профессиональное образование, ISBN: 978-5-7695-4962-5

2. J. Kohl, and C. Neuman, "The Kerberos Network Authentication Service (V5) RFC 1510, September 1993.

3. Denning, Dorothy E.; Sacco, Giovanni Maria (1981). "Timestamps in key distributed protocols". Communication of the ACM 24 (8): 533-535.

# О ПОЛЯРИЗАЦИИ ИСТОЧНИКОВ БЕРНУЛЛИ СЛУЧАЙНЫМИ ЛИНЕЙНЫМИ ПРЕОБРАЗОВАНИЯМИ

**Пантелеев П.А. (Московский Государственный Университет)**
*panteleev@intsys.msu.ru*

В недавней работе Э. Арикана [1] построен класс эффективно кодируемых/декодируемых кодов, достигающих границы Шеннона для двоичного симметричного канала. По существу, как было отмечено в [2], идея построения полярных кодов опирается на *феномен поляризации* дискретных вероятностных источников, состоящий в том, что после применения некоторого специально подобранного преобразования к последовательности $X_1, \ldots, X_n$ двоичных случайных величин получается последовательность двоичных случайных величин $Y_1, \ldots, Y_n$, которую можно условно разбить на две компоненты — «случайную» $Y_{i_1}, \ldots, Y_{i_m}$ и «детерминированную» $Y_{j_1}, \ldots, Y_{j_k}$. Энтропия случайной компоненты близка к максимально возможной, а детерминированная компонента почти однозначно восстанавливается по случайной. Данное обстоятельство позволяет использовать такие преобразования для порождения равномерно распределенных псевдослучайных последовательностей [2]. В настоящей работе изучается феномен поляризации для случайных линейных преобразований.

Для произвольных дискретных случайных величин $U$ и $V$ энтропию $U$ будем обозначать через $H(U)$, а условную энтропию $U$ при условии $V$ через $H(U \mid V)$. Рассмотрим произвольную последовательность двоичных случайных величин $\mathbf{X} = X_1, X_2, \ldots$, которую мы будем называть *источником*. Положим $\mathbf{X}^{(n)} = (X_1, \ldots, X_n)$. Если существует предел

$$\lim_{n\to\infty} \frac{1}{n} H\left(\mathbf{X}^{(n)}\right),$$

то будем называть его *энтропией источника* $\mathbf{X}$, и обозначать $h(\mathbf{X})$. Важным частным видом вероятностного источника является *источник Бернулли*, у которого все случайные величины $X_i$ независимы в совокупности и имеют одинаковое распределение Бернулли с параметром $p$. Легко видеть, что для любого такого источника $\mathbf{X}$ энтропия существует и равна $h(\mathbf{X}) = -p\log_2 p - (1-p)\log_2(1-p)$.

Пусть $\mathbb{F}_2^n$ — множество всех двоичных векторов длины $n$. Зададим произвольную последовательность отображений $g_n \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, $n \in \mathbb{N}$,

преобразующих случайный двоичный вектор $\mathbf{X}^{(n)} = (X_1, ..., X_n)$ в случайный двоичный вектор $\mathbf{Y}^{(n)} = (Y_1, ..., Y_n) = g_n(\mathbf{X}^{(n)})$. Для каждого $\varepsilon > 0$ рассмотрим величину

$$h_\varepsilon(\mathbf{Y}^{(n)}) = \frac{1}{n}|\{i \in \{1, \ldots, n\} \mid H\left(Y_i \mid \mathbf{Y}^{(i-1)}\right) > 1 - \varepsilon\}|.$$

Пусть $i_1, \ldots, i_m$ есть в точности все индексы $i \in \{1, \ldots, n\}$ для которых выполняется $H\left(Y_i \mid \mathbf{Y}^{(i-1)}\right) > 1 - \varepsilon$. Тогда компонента $Y_{i_1}, \ldots, Y_{i_m}$ случайного вектора $\mathbf{Y}^{(n)}$ имеет энтропию $H\left(Y_{i_1}, \ldots, Y_{i_m}\right)$, отличающуюся от максимально возможной не более чем в $(1 - \varepsilon)$ раз, а величина $h_\varepsilon(\mathbf{Y}^{(n)})$ есть доля этой компоненты в $\mathbf{Y}^{(n)}$.

Скажем, что последовательность отображений $g_{i_1}, g_{i_2}, \ldots$ *поляризует* источник $\mathbf{X}$ если для сколь угодно малого $\varepsilon > 0$ выполняется

$$\lim_{k \to \infty} h_\varepsilon(\mathbf{Y}^{(i_k)}) = h(\mathbf{X}).$$

Как показано в [1,2], если в качестве преобразования $g_{2^k}$ берется умножение на $2^k \times 2^k$-матрицу $\mathbf{G}_k = \left(\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right)^{\otimes k}$, где $\otimes k$ — есть $k$-ая кронекерова степень матрицы, то последовательность $g_1, g_2, g_4, \ldots$ поляризует произвольный источник Бернулли.

Рассмотрим ансамбль $\mathcal{A}_n$ случайных двоичных $n \times n$-матриц такой, что все матрицы в нем равновероятны.

**Теорема.** *Пусть $\mathbf{A}_n$ — случайная матрица из ансамбля $\mathcal{A}_n$, а $\mathbf{X}$ — произвольный источник Бернулли. Тогда для любого сколь угодно малого $\delta > 0$ и $\varepsilon_n = o(1)$ выполняется*

$$\lim_{n \to \infty} \Pr\left[|h(\mathbf{X}) - h_{\varepsilon_n}(\mathbf{A}_n \mathbf{X}^{(n)})| < \delta\right] = 1.$$

Таким образом, при растущем $n$ почти все линейные преобразования поляризуют источники Бернулли.

**Литература**

1. Arikan E. Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels // IEEE Transactions on Information Theory. – Vol. 55, Issue 7, 2009, p. 3051–3073.

2. Abbe E. Randomness and dependencies extraction via polarization // Information Theory and Applications Workshop – La Jolla, CA, 2011, p. 1–7.

## БЫСТРОЕ УМНОЖЕНИЕ МАТРИЦ НАД ПОЛЕМ ИЗ ДВУХ ЭЛЕМЕНТОВ

**Чечулина К.А. (Москва, МГУ им. М.В. Ломоносова, механико-математический факультет, кафедра Математической теории интеллектуальных систем)**
*chechulina.karina@gmail.com*

В данной работе речь пойдет о блочных алгоритмах умножения матриц над полем из двух элементов и о параллельных вариантах этих алгоритмов.

Для умножения над полем из двух элементов матрицы $D$ размера $N \times n_1$ на блок $M$ размера $n_1 \times n$ , представленный построчно машинными словами, используется алгоритм "четырех русских". Кратко опишем алгоритм.

Разобьем матрицу $D$ на вертикальные блоки с горизонтальным размером $k$. На первом этапе алгоритма составляем $\frac{n_1}{k}$ массивов длины $2^k$, в каждом из которых хранятся произведения всевозможных битовых строк длины $k$ на соответствующую часть блока $M$. Благодаря тому, что битовые строки упорядочены в лексикографическом порядке, каждый элемент получается сложением двух предшествующих элементов из данного массива. С учетом того, что строки блока $M$ являются машинными словами, их сложение занимает одну операцию. Таким образом, на создание всех массивов понадобится $\frac{n_1}{k} \times 2^k$ сложений по модулю 2. На втором этапе умножаем построчно матрицу $D$ на блок $M$, складывая машинные слова, соответствующие каждому блоку из $k$ бит, который берется остатком от деления машинного слова на $2^k$, а само машинное слово заменяется целой частью от того же деления. При такой реализации для умножения одной строки на блок $M$ потребуется $\frac{n_1}{k}$ сложений. Получение значения, хранящегося в каждом блоке, требует две операции, и общее число операций в программе не превысит $\frac{n_1}{k} \times 2^k + 2 \times N \times \frac{n_1}{k}$. Написанные и протестированные программы подтвердили теоретические оценки сложности.

Рассмотрим параллельные варианты данного алгоритма. Сначала остановимся на параллельной программе умножения матриц машинных слов, рассчитанной на машины с общей памятью на все ядра. В таком случае матрицу $D$ делим на равные вертикальные блоки, ядру будет соответствовать некоторое количество столбцов. Далее каждое ядро умножает свою часть матрицы $D$ на соответ-

ствующую часть столбца $M$, получая столбец - произведение. Все эти столбцы необходимо сложить. Время работы оценивается величиной $\frac{2\times N\times n_1}{k\times nit} + N$ операций, где $nit$ — количество ядер. Протестированные на СКИФ МГУ программы показали, что на машинах с общей памятью достигается почти 100% ускорение, что соответствует теоретическим оценкам, поскольку не зависящее от $nit$ слагаемое мало.

Второй вариант программы рассчитан на вычислительные узлы с разделенной памятью. Матрица $D$ делится между вычислительными узлами горизонтально, то есть каждый узел получает примерно одинаковое количество строк, которые умножает на столбец $M$, получая соответствующие строки матрицы-произведения. Далее, с уменьшенным параметром $N$, каждый узел делит между своими ядрами абсолютно так же, как в предыдущем варианте программы (вертикально). После этого все результаты надо циклически переслать остальным вычислительным узлам. В таком случае каждый вычислительный узел тратит $\frac{n_1\times 2^k}{nit\times k} + \frac{N\times n_1}{nit\times k\times p}$ операции на нахождение своей части матрицы, где $p$ - количество вычислительных узлов. Если рассматривать только вычислительную часть программы, то достигается 100% ускорение как в зависимости от числа вычислительных узлов, так и в зависимости от числа ядер на узле. Однако если учитывать и пересылки, то использование большого числа вычислительных узлов не является эффективным, поскольку на сбор информации понадобится значительно больше времени, чем на вычисление. Оптимальным параметром $p$ при больших $N$ является $p = 40$.

Пусть требуется вычислить $L^T \times M$, где $L, M \in \mathbb{F}(N \times s \times n)$. Разделим левый блок $L$ по столбцам на блоки меньшего размера $k$. В каждом блоке записаны слова из $k$ бит, поэтому различных строк не более $2^k$. Обозначим блоки через $L_i$ и будем умножать $L_i^T$ на блок $M$, в каждой строке которого $s$ машинных слов. На первом этапе проходим блок $L_i$ слева направо все столбцы и выбираем разные. Если $j$-ый столбец уже встречался на месте $t$, то прибавляем $j$-ую строку блока $M$ к $t$-ой строке. Далее рассматриваем только различные столбцы блока $L_i$. На втором этапе получившиеся матрицы размера не более чем $2^k$ перемножаем. Число операций на этом этапе равно $n \times s^2 \times 2^k$. Выбирая $k = \log_2 N - \log_2 \log_2 N$, получаем общую оцен-

ку сложности работы программы $3 \times \frac{N \times n \times s^2}{\log_2 N - \log_2 \log_2 N}$. Проведенные тесты подтвердили теоретические оценки.

Первые два способа распараллеливания написаны для машин с общей памятью на все ядра.

Первый способ заключается в том, что каждому ядру выделяется равное количество вертикальных блоков $L_i$. Однако всем ядрам придется скопировать себе весь блок $M$. Как и в первом варианте распараллеливания умножения матриц, блок $M$ передается циклически. Далее каждое ядро будет полностью повторять действия однопроцессорной версии, только с уменьшенной матрицей $L_i$. Каждое ядро получает некоторые строки итогового произведения, соответствующие высланным ему столбцам матрицы $L$, которые оно записывает в матрицу-произведение. В итоге необходимо $\frac{s \times N}{nit} + s \times N$ пересылок машинных слов, $\frac{3 \times s^2 \times N \times n}{nit \times (\log_2 N - \log_2 \log_2 N)}$ операций и $\frac{s^2 \times s}{nit}$ перезаписей машинных слов на формирование результата. Эффективность распараллеливания составляет около 70%. Это обусловлено тем, что есть не зависящее от $nit$ слагаемое.

Второй вариант отличается от первого тем, что оба столбца делятся на части, и каждое ядро получает соответствующую пару блоков (часть столбцов $L$ и часть столбцов $M$), которые он перемножает, получая соответствующие части матрицы-произведения. При такой реализации необходимо разложить число ядер $nit$ на множители $nit_1$ и $nit_2$, первый столбец будем делить на $nit_1$ частей, а второй - на $nit_2$. Тогда необходимо $\frac{s \times N}{nit_1} + \frac{s \times N}{nit_2}$ перезаписей машинных слов, $\frac{3 \times s^2 \times N \times n}{(nit_1 \times nit_2 \times (\log_2 N - \log_2 \log_2 N))}$ операций, и затем $\frac{s^2 \times n}{nit_1 \times nit_2}$ перезаписей для сбора результатов. Выгодно брать $nit_1 \approx nit_2$. Результаты оказались примерно такими же, ускорение составило около 75%.

Третий вариант программы рассчитан на вычислительные узлы с разделенной памятью. У каждого узла хранится соответствующая ему часть первого столбца, и часть второго столбца, каждая часть примерно $\frac{N}{\sqrt{p}}$ строк, где $p$ — количество вычислительных узлов. Далее на каждом вычислительном узле происходит распараллеливание между ядрами по первому варианту. В итоге после умножения каждый вычислительный узел будет владеть матрицей размера $(s \times n) \times s$. Эти матрицы необходимо сложить, чтобы получить матрицу - произведение. Сложение будет происходить циклически, каждый столбец складывается по очереди. Оценим сложность такого алгоритма.

Необходимо $\frac{3\times s^2\times N\times n}{p\times nit\times(\log_2 N-\log_2\log_2 N)}$ операций и $\frac{2\times(p-1)\times s^2\times n}{p}$ пересылок на сбор информации. При $p$ меньше 20 достигается примерно 70% ускорение по $nit$ и 85% ускорение по $p$. Однако, при больших значениях $p$ (около 50—60) ускорение уменьшается, и составляет примерно 60% в зависимости от количества вычислительных узлов.

Итак, использование описанных алгоритмов позволяет вычислить произведение матриц в 32 раза быстрее при непараллельной версии программы (на оборудовании СКИФ МГУ). Ускорение же при распараллеливании составляет около 200 раз для умножения матриц машинных слов и примерно 400 раз при вычислении скалярного произведения столбцов.

**Литература**

1. Coppersmith D, "Solving homogeneous linear equations over GF(2) via Block Wiedemann Algorithm Math. of Comp., Vol. 62, No. 205, (Jan 1994), pp. 333-350

2. Богачев К. Ю., "Основы параллельного программирования Бином. Лаборатория знаний, 2003

# Секция “Дискретная математика и математическая кибернетика”

# ОБ ЭФФЕКТИВНОСТИ АЛГОРИТМОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ НЕКОТОРЫХ КЛАССОВ БУЛЕВЫХ ФУНКЦИЙ

**Архипова А.Н. (Московский Государственный Университет имени М.В.Ломоносова)**
*a.n.arkhipova@yandex.ru*

В работе [1] рассматривается Probably Approximately Correct Model (PAC - модель) для изучения эффективности алгоритмов машинного обучения, в которой предполагается задание распределения на множестве примеров и целью познающего является нахождение гипотезы, аппроксимирующей целевую функцию с заранее заданной точностью. В работе доказывается неэффективность работы алгоритма Персептрон на классе линейных пороговых функций и эффективность на классе вложенных функций. При доказательстве первого факта использовались определение и свойства функции Хастада, построение которой можно найти в статье [2].

Таким образом, косвенно утверждается непринадлежность функции Хастада классу вложенных функций. В данной работе получено прямое доказательство данного факта.

**Утверждение:** $F(f)$ - *функция Хастада не лежит в множестве* $NF_n$, *где* $NF_n$- *множество вложенных функций от* $n$ *переменных,* $n = 2^m, m \geq 3$.

Использование класса пороговых функций для определения эффективности работы различных алгоритмов приводит к необходимости изучения некотрых их свойств и взаимосвязей.

Пусть

$$\rho^{'}(f^{'}, f^{''}) = \min_{l_{\vec{\omega}^{'},\sigma^{'}} \to f^{'}, l_{\vec{\omega}^{''},\sigma^{''}} \to f^{''}} \rho(l^{'}, l^{''})$$

где $l_{\vec{\omega},\sigma} \to f(x_1, ..., x_n)$ - линейная форма, задающая пороговую функцию $f$ (линейной формой называется функция вида $l_{\vec{\omega},\sigma}(x_1, ..., x_n) = \sum_{i=1}^{n} x_i\omega_i - \sigma, \omega_i \in Z, i = 1, ..., n; \sigma \in Z$).

**Утверждение:** *Для любой тройки функций вида*

$$f_1 : x_1 + 2x_2 + 4x_3 + ... + 2^{i-1}x_i \geqslant \sum_{j=1}^{i-1} 2^{j-1}, i \geq 4$$

$$f_2 : x_1 \geqslant 1$$

$$f_3 : x_i \geqslant 1$$

*нарушается нераверство треугольника для функции* $\rho^{'}$, *т.е.*

$$\rho^{'}(f^{'}, f^{''}) > \rho^{'}(f^{'}, f^{'''}) + \rho^{'}(f^{''}, f^{'''})$$

**Благодарности**

Благодарю моего научного руководителя Ирматова Анвара Адхамовича за ценные рекомендации и советы.

**Литература**

1. Rocco A.Servedio. On PAC Learning Using Winnow, Perceptron and a Perceptron-Like Algorithm.

2. Johan Hastad. On the size of weigts for threshold gates.

# О СПЕКТРАХ КЛАССОВ ПОСТА

**Блохина Г.Н., Кудрявцев В.В.**

Изучаются длины базисов классов Поста булевских функций. Множество всех длин базисов такого класса называется его спектром. Основным результатом является нахождение всех спектров классов Поста.

**Введение**

В 1921 г. Э. Пост опубликовал работу [1], в которой описал множество всех замкнутых относительно суперпозиции классов булевских функций, называемых теперь классами Поста.

Он показал, что все эти классы конечно-порожденные и их счетное число. Им построена структура по включению, образованная этими классами. Работа Э. Поста долго была малоизвестной.

В 1966 г. появилась монография [2], в которой было переизложено доказательство этих результатов Э. Поста в более краткой форме. В ней были также вычислены порядки классов Поста, то есть найдены базисы этих классов, содержащие наименьшее число переменных.

Позже в [3] были найдены предикаты, зависящие от наименьшего числа переменных, классами сохранения которых являются предикатные классы Поста.

Здесь мы вводим еще одну характеристику классов Поста, называемую спектром. Спектром класса Поста является множество всех длин базисов его. Нашей целью является описание спектров всех классов Э. Поста, которая здесь достигается. Мы показываем, что спектрами являются множества {1}, {2}, {3}, {1,2}, {2,3}, {1,2,3}, {1,2,3,4} и только они.

**1. Основные понятия и результаты**

Пусть $E_2 = \{0,1\}$, $\mathbb{N} = \{1,2,\ldots,n,\ldots\}$ и $U = \{u_1, u_2, \ldots, u_k, \ldots\}$ — алфавит переменных $u_n$, принимающих значения из $E_2$, $n \in \mathbb{N}$. Обозначим через $C_1$ класс всех функций $f(u_{i_1}, u_{i_2}, \ldots, u_{i_m})$, $m \in \mathbb{N}$, значения аргументов которых и самих функций из $E_2$. Эти функции называются булевскими, б. функциями или просто функциями. Далее для обозначения переменных будут иногда использоваться мета-символы $x, y, z$, возможно, с индексами.

В классе $C_1$ обычным образом вводятся операции переименования переменных и подстановки одних функций в другие вместо их переменных. Эти операции называются операциями суперпозиции.

С их помощью строятся формулы из функций. Эти формулы реализуют функции, которые называют суперпозициями тех функций, из которых строятся формулы.

Если $M \subseteq C_1$, то через $[M]$ обозначим все суперпозиции, реализуемые формулами, построенными с помощью функций из $M$. Оператор $[M]$ является замыканием $M$ и для него выполнено:

1. $[M] \supseteq M$;

2. если $M_1 \supseteq M_2$, то $[M_1] \supseteq [M_2]$;

3. $[[M]] = [M]$.

Говорим, что $M$ замкнуто, если $[M] = M$.

Если $M$ — замкнуто и $M_1 \subseteq M$, то при $[M_1] = M$ говорим, что $M_1$ — полно в $M$.

В случае, когда $M_1$ конечно, говорим, что $M$ является конечнопорожденным. Полное в $M$ множество $M_1$ называется базисом в $M$, если для любого $M_2$ такого, что $M_2 \subset M_1$, выполнено $[M_2] \neq M$, число элементов в $M_1$ называем длиной базиса $M_1$. Для обозначения мощности множества $T$ используем далее символ $|T|$.

Для замкнутого $M$ обозначим через $\Sigma(M)$ множество всех замкнутых классов $M_1$ таких, что $M_1 \subseteq M$.

Решетка, которую образует множество $\Sigma(M)$ с отношением включения между его элементами, обозначается через $\overline{\Sigma(M)}$ и представляется в виде графа.

Вершинами этого графа являются классы функций из $\Sigma(M)$, вложение которых друг в друга обозначается ребром, ориентированным от более широкого класса $M$ к классу $M'$, вложенному в него, если нет такого $M''$, что $M \subset M'' \subset M'$.

Граф располагают вертикально, когда указанная ориентация ребер идет сверху вниз и потому возможна замена стрелок в графе на дуги и отрезки.

К числу основных задач для $M$ относится задача о строении графа $\overline{\Sigma(M)}$. Она оказалась связанной с задачей о полноте.

Задача о полноте для $M$ состоит в описании всех решений уравнения $[X] = M$, $X \subseteq M$, относительно $X$.

Решение задачи о полноте в $M$ получается с помощью понятия критериальной системы.

Подмножество $\Theta(M) \subseteq \Sigma(M)$ называется критериальной системой для M (к. системой), если всякое множество $M_1 \subseteq M$ является полным точно тогда, когда для любого $Q$ из $\Theta(M)$ выполнено

$M_1 \not\subseteq Q$. Примером к. системы является $\Sigma(M)\backslash M$, если $\Sigma(M)\backslash M \neq \emptyset$. Вместе с тем ясно, что если $\Theta(M)$ — к. система и для некоторых ее элементов $Q_1$ и $Q_2$ выполнено $Q_1 \subseteq Q_2$, то $\Theta(M)\backslash\{Q_1\}$ также к. система. Таким образом, при рассмотрении к. систем как инструмента для решения задачи о полноте естественно требовать, чтобы к. система была неизбыточной. Это может быть осуществлено с помощью понятия предполного класса (п. класса).

Назовем замкнутое множество $Q$ из $\Sigma(M)\backslash M$ п. классом, если для любой функции $f$ из $M\backslash Q$ выполнено $[Q \bigcup \{f\}] = M$. Нетрудно видеть, что каждый п. класс входт в любую к. систему $\Theta(M)$. Обозначая через $\Sigma_\pi(M)$ класс всех п. классов, имеем $\Theta(M) \supseteq \Sigma_\pi(M)$.

Может случиться, что для $M$ множество $\Sigma_\pi(M)$ является к. системой, тогда задача о полноте в $M$ становится эквивалентной указанию $\Sigma_\pi(M)$.

Опишем все классы Поста. Их множество исчерпывается списком $C_i$, $A_i$, $D_j$, $L_k$, $O_l$, $S_r$, $P_r$, $F_s^\mu$, $F_s^\infty$, где $i$ =1,2,3,4; $j$ =1,2,3; $k$ =1,2,3,4,5; $l$ =1,2,3,4,5,6,7,8,9; $r$ =1,3,5,6; $s$ =1,2,. . . ; $\mu$ =2,3,. . .

Класс $C_1$ содержит все б. функции; $C_2$ состоит из всех функций $f(x_1,\ldots,x_n)$ таких, что $f(0,\ldots,0)=0$; $C_3$ — из всех функций таких, что $f(1,\ldots,1)=1$; $C_4 = C_2 \bigcap C_3$.

Классы $C_1$,$C_2$,$C_3$,$C_4$ образуют группу классов Поста типа $C$.

Функция $f(x_1,\ldots,x_n)$ называется монотонной, если для любых наборов $\alpha = (a_1,\ldots,a_n)$ и $\beta = (b_1,\ldots,b_n)$ таких, что $a_i \leq b_i$ для всех $i$ =1,2,. . . ,$n$, что обозначаем $\alpha \leq \beta$, выполнено $f(\alpha) \leq f(\beta)$.

Класс $A_1$ состоит из всех монотонных функций, $A_2 = C_2 \bigcap A_1$; $A_3 = C_3 \bigcap A_1$; $A_4 = A_2 \bigcap A_3$.

Классы $A_1$, $A_2$, $A_3$, $A_4$ называем классами типа $A$.

Функция $f^*(x_1,\ldots,x_n) = \bar{f}(\bar{x}_1,\ldots,\bar{x}_n)$ называется двойственной к $f(x_1,\ldots,x_n)$. Если $f = f^*$, то $f$ называем самодвойственной.

Класс $M^*$, состоящий из всех функций $f^*$, двойственным функциям $f$ из $M$, называем двойственным к $M$.

Класс $D_3$ состоит из всех самодвойственных функций; $D_1 = D_3 \bigcap C_4$; $D_2 = D_3 \bigcap A_1$.

Классы $D_1$, $D_2$, $D_3$ называем классами типа $D$.

Класс $L_1$ состоит из всех линейных функций, то есть таких $f(x_1,\ldots,x_n)$, что $f(x_1,\ldots,x_n) = \sum\limits_{i=1}^{n} a_i x_i + d \ (mod 2)$. $L_2 = L_1 \bigcap C_2$; $L_3 = L_1 \bigcap C_3$; $L_4 = L_1 \bigcap C_4$; $L_5 = L_1 \bigcap D_3$.

Классы $L_1$, $L_2$, $L_3$, $L_4$, $L_5$ называем классами типа $L$.

Класс $O_9$ состоит из функций, существенно зависящих не более, чем от одного переменного; $O_8 = O_9 \bigcap A_1$; $O_4 = O_9 \bigcap D_3$; $O_5 = O_9 \bigcap C_2$; $O_6 = O_9 \bigcap C_3$; $O_1 = O_5 \bigcap O_6$; $O_7 = \{0,1\}$; $O_2 = O_5 \bigcap O_7$; $O_3 = O_6 \bigcap O_7$.

Класс $S_6$ состоит из всех функций вида $x_1 \vee x_2 \vee \ldots \vee x_n$ и констант 0 и 1; $S_3 = S_6 \bigcap C_2$; $S_5 = S_6 \bigcap C_3$; $S_1 = S_5 \bigcap S_3$.

Классы $S_1$, $S_3$, $S_5$, $S_6$ называем классами типа $S$.

Класс $P_6$ состоит из всех функций вида $x_1 \cdot x_2 \cdot \ldots \cdot x_n$ и констант 0 и 1; $P_5 = P_5 \bigcap C_2$; $P_3 = P_6 \bigcap C_3$; $P_1 = P_5 \bigcap P_3$.

Классы $P_1$, $P_3$, $P_5$, $P_6$ называем классами типа $P$.

Функция $f$ удовлетворяет условию $a^\mu$, если любые $\mu$, $\mu \geq 2$, наборов, на которых $f$ равна 0, имеют общую единичную координату, равную нулю.

Аналогично с заменой 0 на 1 вводится условие $A^\mu$.

Класс $F_4^\mu$ состоит из всех функций со свойством $a^\mu$; $F_1^\mu = C_4 \bigcap F_4^\mu$; $F_3^\mu = A_1 \bigcap F_4^\mu$; $F_2^\mu = F_1^\mu \bigcap F_3^\mu$. $F_8^\mu$ состоит из всех функций со свойством $A^\mu$; $F_5^\mu = C_4 \bigcap F_8^\mu$; $F_7^\mu = A_3 \bigcap F_8^\mu$; $F_6^\mu = F_5^\mu \bigcap F_7^\mu$.

Функция удовлетворяет условию $a^\infty$, если все наборы, на которых она равна нулю, имеют общую нулевую координату.

Аналогично с заменой 0 на 1 вводится свойство $A^\infty$.

Класс $F_4^\infty$ состоит из всех функций со свойством $a^\infty$; $F_1^\infty = C_4 \bigcap F_4^\infty$; $F_3^\infty = A_1 \bigcap F_4^\infty$; $F_2^\infty = F_1^\infty \bigcap F_3^\infty$. $F_8^\infty$ состоит из всех функций со свойством $A^\infty$; $F_5^\infty = C_4 \bigcap F_8^\infty$; $F_7^\infty = A_3 \bigcap F_8^\infty$; $F_6^\infty = F_5^\infty \bigcap F_7^\infty$.

Множество перечисленных классов обозначим через $\Delta$. Э. Постом установлено следующее утверждение [1].

**Теорема 1.** *Имеют место положения:*

*а)* $\Delta = \Sigma(C_1)$;

*б)* $\Sigma(C_1)$ *имеет вид, как на рис. 1.*

*Этот граф будем называть диаграммой Поста.*

Из этой теоремы вытекает справедливость следующего утверждения.

**Теорема 2.** *Имеют место следующие положения:*

*а) для любого* $Q$ *из* $\Sigma(C_1) \setminus \{O_1, O_2, O_3\}$ $\Sigma_\pi(Q)$ *является к. системой и* $|\Sigma_\pi(Q)| \leq 5$;

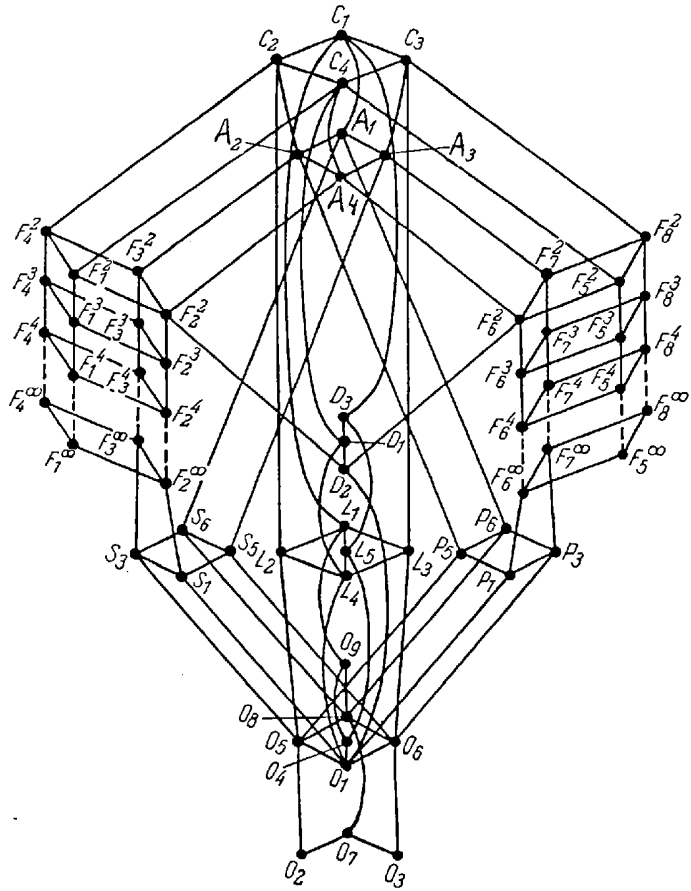



Рис. 1: Диаграмма Поста

*б) каждый класс $Q$ из $\Sigma(C_1)$ является конечно-порожденным, и для базиса $Б$ в $Q$ выполнено $|Б|\leq 5$.*

Обозначим через $spQ$ множество всех длин базисов в $Q$ и назовем его спектром. Нашей задачей является нахождение спектров всех классов Поста.

Справедливо следующее основное утверждение.

**Теорема 3.** *Для любого $Q$ из $\Sigma(C_1)$ справедливы положения:*

*а) $spQ=\{1\}$, если $Q\in\{D_2, L_4, F_1^\infty, F_5^\infty, F_6^\infty, F_2^\infty, P_1, S_1, O_1, O_2, O_3, O_4\}\cup(\bigcup_{\mu>2}^{\infty}\{F_2^\mu\})\cup(\bigcup_{\mu>2}^{\infty}\{F_6^\mu\})$;*

*б) $spQ=\{2\}$, если $Q\in\{P_3, P_5, S_3, S_5, O_5, O_6, O_7, O_9, F_3^\infty, F_7^\infty\}\cup(\bigcup_{\mu=2}^{\infty}\{F_3^\mu\})\cup(\bigcup_{\mu=2}^{\infty}\{F_7^\mu\})$;*

*в) $spQ=\{3\}$, если $Q\in\{P_6, S_6, O_3\}$;*

*г) $spQ=\{1,2\}$, если $Q\in\{C_4, D_1, D_3, L_3, L_2, L_5, F_2^2, F_6^2, F_4^\infty F_8^\infty\}\cup(\bigcup_{\mu=2}^{\infty}\{F_1^\mu\})\cup(\bigcup_{\mu=2}^{\infty}\{F_4^\mu\})\cup(\bigcup_{\mu=2}^{\infty}\{F_5^\infty\})\cup(\bigcup_{\mu=2}^{\infty}\{F_8^\mu\})$;*

*д) $spQ=\{2,3\}$, если $Q\in\{L_1, A_2, A_3\}$;*

*е) $spQ=\{3,4\}$, если $Q=A_1$;*

*ж) $spQ=\{1,2,3\}$, если $Q\in\{C_2, C_3\}$;*

*з) $spQ=\{1,2,3,4\}$, если $Q=C_1$;*

*и) других спектров для $Q$ из $\Sigma(C_1)$, кроме указанных в пунктах а)–з), не существует.*

## Литература


1. Post E. Two-valued iterative systems.Prinston,1941.

2. Яблонский С.В.,Гаврилов Г. П., Кудрявцев В. Б. Функции алгебры логики и классы Поста. М.: Физматлит, 1966.

3. Блохина Г. Н. О предикатном описании классов Поста Дискретный анализ. Вып. 16. Новосибирск, 1970.

4. Кудрявцев В. В. О функциональной системе пучков логических функций , Фундаментальная и прикладная математика. Т. 5, вып. 1. М.: Изд-во МГУ.

# О КОНЕЧНОЙ-ПОРОЖДЕННОСТИ ИСЧИСЛЕНИЯ ВЫСКАЗЫВАНИЙ С ПРОИЗВОЛЬНЫМИ ОПЕРАЦИЯМИ ВЫВОДА

**Боков Г.В. (МГУ им. М.В. Ломоносова)**
*bokovgrigoriy@gmail.com*

Важным свойством классического исчисления высказываний [1] является существование конечного множества аксиом, из которых выводимы все тавтологии этого исчисления. Данное свойство называют конечно-порожденностью исчисления. При расширении понятия исчисления обычно требуется, чтобы данное свойство сохранялось. Так в 1949 г. Л. Хенкин [2] показал, что расширенный фрагмент исчисления высказываний, содержащий классическую импликацию, конечно-порожден относительной операции *modus ponens*: если выводимо $A$ и $A \to B$, то $B$ выводимо. В данной работе вводиться в рассмотрение расширенный фрагмент исчисления высказываний с произвольными модусными операциями [3] и доказываются необходимые и достаточные условия конечно-порожденности такого исчисления.

Пусть $A$ — некоторое множество и $U = \{u_1, u_2, ..., u_n, ...\}$ — счетный алфавит переменных $u_n$, значениями которых являются элементы $a$ из $A$. Обозначим через $P_A$ множество всех функций $f(u_{i_1}, ..., u_{i_n})$ со значениями в $A$, где $i_j < i_{j'}$ при $j < j'$, $j, j' = 1, ..., n$, $n \in \mathbb{N}$. Эти функции являются отображениями вида $f : A^n \to A$, где $A^n = \underbrace{A \times ... \times A}_{n}$. Такие функции иногда называют функциями $|A|$-значной логики.

Множество переменных $U$ будем интерпретировать, как переменные высказывания, а множество $A$ — как значения, которые могут принимать высказывания. В такой интерпретации функции из множества $P_A$ можно рассматривать как логические связки над высказываниями. Если $|A| = 2$, то элементами множества $A$ будут 1 и 0, которые интерпретируются стандартным образом, как истина и ложь. Для произвольного множества $A$ обобщим понятие истины и лжи следующим образом. Пусть $\rho$ — произвольный предикат на $A$, т.е. отображение $\rho : A \to E_2$, где $E_2 = \{0, 1\}$. Тогда множество $\mathbb{T} = \{a \in A \mid \rho(a) = 1\}$ будем интерпретировать, как множество истинных значений, а множество $\mathbb{F} = \{a \in A \mid \rho(a) = 0\}$ — как мно-

жество ложный значений. Пару $(\mathbb{T}, \mathbb{F})$, порожденную предикатом $\rho$, будем называть *истинностным разбиением* множества $A$.

Каждый предикат $\rho$ порождает естественный гомоморфизм $\mathrm{nat}_\rho$ множества $P_A$ в множество всех функций двузначной логики $P_2$ [4]. Этот гомоморфизм каждой функции $f(x_1, ..., x_n)$ из $P_A$ сопоставляет такую функцию $f_\rho(x_1, ..., x_n)$ из $P_2$, что для любого набора $\langle\sigma_1, ..., \sigma_n\rangle \in E_2^n$ и любого $\sigma_0 \in E_2$ выполнено

$$f_\rho(\sigma_1, ..., \sigma_n) = \sigma_0 \Leftrightarrow f(\hat{\sigma}_1, ..., \hat{\sigma}_n) \subseteq \hat{\sigma}_0$$

где $\hat{\sigma}_i = \mathbb{T}$, при $\sigma_i = 1$, и $\hat{\sigma}_i = \mathbb{F}$, при $\sigma_i = 0$, $i = 0, 1, ..., n$.

Пусть $\Sigma \subseteq P_A$ — конечное множество функций и $X \subseteq U$ — множество переменных, тогда обозначим через $\Phi_\Sigma(X)$ множество всех формул над логическими связками из $\Sigma$ и множеством переменных $X$. Когда $X = U$ множество $\Phi_\Sigma(X)$ будем для краткости обозначать через $\Phi_\Sigma$. Каждой формуле $\mathfrak{F} \in \Phi_\Sigma$ можно однозначно сопоставить функцию $f_\mathfrak{F} \in P_A$ [4]. В этом случае говорят, что формула $\mathfrak{F}$ выражает функцию $f_\mathfrak{F}$. Формулу $\mathfrak{F}$ из $\Phi_\Sigma$ будем называть *тавтологией*, т.е. тождественно истинной, относительно истинностного разбиения $(\mathbb{T}, \mathbb{F})$, если $f_\mathfrak{F}(i_1, ..., i_m) \in \mathbb{T}$, при любых значениях $i_1, ..., i_m$ из $A$, где $m$ — это арность функции $f_\mathfrak{F}$. Обозначим через Th множество всех тавтологий в $\Phi_\Sigma$.

Определим понятие модусной операции. Пусть $\mathfrak{F}_0, \mathfrak{F}_1, ..., \mathfrak{F}_m$ различные формулы из $\Phi_\Sigma(\{x_1, ..., x_n\})$, тогда набор $\langle\mathfrak{F}_1, ..., \mathfrak{F}_m; \mathfrak{F}_0\rangle$ задает модусную операцию на $\Phi_\Sigma$, определенную схемой:

$$\frac{\mathfrak{F}_1(x_1, ..., x_n), ..., \mathfrak{F}_m(x_1, ..., x_n)}{\mathfrak{F}_0(x_1, ..., x_n)}$$

Классическим примером модусной операци является операция *modus ponens*:

$$\frac{x_1, x_1 \to x_2}{x_2}$$

Множество всех модусных операций на $\Phi_\Sigma$ обозначим через $\mathcal{M}_\Sigma$. Нас будут интересовать не все модусные операции, а лишь те $\omega \in \mathcal{M}_\Sigma$, которые тавтологии переводят в тавтологии $\omega : \mathrm{Th} \to \mathrm{Th}$. Такие операции назовем допустимыми [5] на Th. Множество всех допустимых на Th операций обозначим через $O_{\mathrm{Th}}$.

Множество тавтологий Th и множеств допустимых на Th операций $\Omega \subseteq O_{\mathrm{Th}}$ образуют алгебраическую систему $(\mathrm{Th}, \Omega)$, которую будем называть *исчислением высказываний*.

Пусть $\Omega \subseteq O_{\mathrm{Th}}$ — произвольное множество допустимых на Th модусных операций, тогда на Th можно определить оператор замыкания, порожденный операциями из $\Omega$ [6]. Этот оператор будем обозначать через $[\cdot]_\Omega$. Для произвольного $M \subseteq \mathrm{Th}$ и $\mathfrak{A} \in \mathrm{Th}$ формулу $\mathfrak{A} \in [M]_\Omega$ назовем выводимой из множества формул $M$ и обозначим это через $M \vdash_\Omega \mathfrak{A}$. Множество тавтологий Th будем называть конечно-порожденным относительно множества операций $\Omega$, если существует такое конечное множество $M \subseteq \mathrm{Th}$, что $[M]_\Omega = \mathrm{Th}$. Исчисление высказываний $(\mathrm{Th}, \Omega)$ конечно-порождено, если множество Th конечно-порождено относительно множества операций $\Omega$.

Пусть $X \subseteq U$ некоторое множество переменных, тогда обозначим через $\mathcal{M}_\Sigma(X) \subseteq \mathcal{M}_\Sigma$ множество всех операций над переменными из $X$:

$$\mathcal{M}_\Sigma(X) = \{\omega \in \mathcal{M}_\Sigma \mid \omega = \langle \mathfrak{F}_1, ..., \mathfrak{F}_m; \mathfrak{F}_0 \rangle \text{ и } \mathfrak{F}_i \in \Phi_\Sigma(X),\ i = 0, 1, ..., m\}$$

Будем говорить, что операция $\omega = \langle \mathfrak{F}_1, ..., \mathfrak{F}_m; \mathfrak{F}_0 \rangle \in \mathcal{M}_\Sigma(\{x_1, ..., x_n\})$ выводима из множества операций $\Omega \subseteq \mathcal{M}_\Sigma$, если существует такое конечное множество тавтологий $M \subseteq \mathrm{Th}$, что для любых формул $\mathfrak{A}_1, ..., \mathfrak{A}_n, \mathfrak{B}_1, ..., \mathfrak{B}_m \in \Phi_\Sigma$ из выполнения условия $\mathfrak{B}_i = \mathfrak{F}_i(\mathfrak{A}_1, ..., \mathfrak{A}_n) \in \mathrm{Th}, i = 1, ..., m$ следует выводимость

$$M,\ \mathfrak{B}_1,\ ...,\ \mathfrak{B}_n \vdash_\Omega \mathfrak{F}_0(\mathfrak{A}_1, ..., \mathfrak{A}_n)$$

Выводимость операции $\omega$ из множества операций $\Omega$ обозначим через $\Omega \vdash \omega$.

Определим несколько классов функций. Функцию $f(x_1, ..., x_n) \in P_A$ назовем *линейной* (*монотонной*) относительно предиката $\rho$, если $f_\rho = \mathrm{nat}_\rho(f) \in P_2$ является линейной (монотонной) булевской функцией. Множество всех линейных и монотонных относительно $\rho$ функций в $P_A$ обозначим соответственно через $L_\rho$ и $M_\rho$. Функцию $f(x_1, ..., x_n, y) \in P_A$ назовем *импликативной* относительно предиката $\rho$, если для функции $f_\rho = \mathrm{nat}_\rho(f) \in P_2$ выполнено

$$f_\rho(\sigma_1, ..., \sigma_n, \sigma_0) = 0 \Leftrightarrow \sigma_0 < \sigma_i,\ i = 1, ..., n,$$

где $\sigma_i \in E_2$. Множество всех импликативных относительно $\rho$ функций в $P_A$ обозначим через $I_\rho$.

**Теорема.** *Для любого множества $A$, любого предиката $\rho$ на $A$ и каждого конечного множества логических связок $\Sigma \subseteq P_A$ существуют такие конечные множества допустимых на* Th *операций $\Omega_L(\Sigma)$, $\Omega_M(\Sigma)$, $\Omega_I(f)$, $f \in \Sigma$, что для произвольного конечного множества операций $\Omega \subseteq O_{\mathrm{Th}}$ исчисление $(\mathrm{Th}, \Omega)$ конечно-порождено тогда и только тогда, когда выполнено хотя бы одно из условий*

1. $\Sigma \subseteq L_\rho$, $1 \in [nat_\rho(\Sigma)]$ *и* $\Omega \vdash \Omega_L(\Sigma)$;
2. $\Sigma \subseteq M_\rho$, $1 \in [nat_\rho(\Sigma)]$ *и* $\Omega \vdash \Omega_M(\Sigma)$;
3. *Найдется такая функция* $f \in [\Sigma] \cap I_\rho$, *что* $\Omega \vdash \Omega_I(f)$;
3. $\mathrm{Th} = \emptyset$.

**Литература**

1. Новиков П.С. Элементы математической логики. - М.: Наука, 1973.

2. Henkin L. Fragments of the propositional calculus. J. Symb.Logic, 14 (1949), 42—82.

3. Циткин А. И. О допустимых правилах интуиционистской логики высказываний, Матем. сб., 102(144):2 (1977)

4. Яблонский С.В. Введение в дискретную математику. — М., Наука, 1986.

5. Минц Г. Е. Допустимые и производные правила, Записки научных семинаров ЛОМИ АН СССР, 8 (1968), 189—191.

6. Кон П. Универсальная алгебра. - М.: Мир, 1968.

# ПОСТРОЕНИЕ СИНХРОНИЗИРУЮЩИХ ДЕРЕВЬЕВ

**Гасанов Э.Э., Дин А.А.**
**(Московский государственный университет)**
*el_gasanov@mail.ru*

В данной работе рассматривается известная проблема синхронизации сигнала, возникающая при производстве электронных схем (чипов). Она состоит в том, чтобы от некоторой точки чипа (источника) до некоторых других точек чипа (стоков) сигнал доходил одновременно. В чипах в качестве источника выступает выход генератора периодических сигналов, определяющих тактовую частоту чипа, а в качестве стоков — управляющие входы регистров, которые определяют состояние чипа в каждый момент времени. Одновременность поступления сигнала гарантирует одномоментность изменения состояния чипа через равные промежутки времени. Эта задача решалась как для чисто производственных алгоритмов [1], так и на модельных объектах [2,3]. В данной работе используется модель, предложенная в [3].

В качестве модели мы будем рассматривать ориентированные деревья с корнем, каждая вершина которых лежит на плоской целочисленной решетке, каждое ребро соединяет две соседние вершины целочисленной решетки (т.е. каждое ребро имеет длину 1, а степень инцидентности каждой вершины не более 4), и все ребра направлены от корня к концевым вершинам, при этом *корень* — это вершина, в которую не входит ни одно ребро, а из *концевых вершин* не исходит ни одного ребра. На рисунке 1 приведен пример такого дерева, причем корень дерева помечен треугольником, а концевые вершины помечены жирными точками. Ориентация ребер на рисунке не приведена, так как после фиксации корня она определяется однозначно.

*Полустепенью исхода вершины* дерева назовем число ребер, исходящих из вершины.

*Задержкой вершины* дерева назовем ее полустепень исхода, *задержкой пути* в дереве — сумму задержек всех вершин пути, а *задержкой до концевой вершины* дерева — задержку пути, ведущего от корня к этой концевой вершине. Такое определение задержки продиктовано технологическими особенностями распространения сигналов в чипах. Понять эти особенности может помочь следующая "водная" интерпретация. Представим, что в каждой вершине стоит

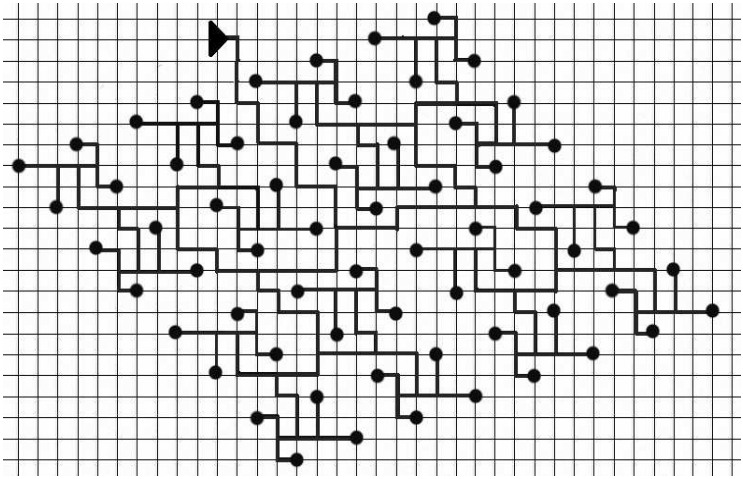


Рис. 1: Пример построенного синхронизирующего дерева для конфигурации $K \in \mathcal{K}_4$

насос, а каждое исходящее ребро это каналы, данным насосом заполняемые. Насос включается, если заполнен канал, соответствующий входному ребру вершины, поэтому насосы, соответствующие концам исходящих ребер включатся только тогда, когда заполнятся каналы, соответствующие данным ребрам. А время заполнения каналов пропорционально их количеству.

*Конфигурацией* назовем множество точек плоской целочисленной решетки, одна из которых называется *источником*, а остальные точки конфигурации называются *стоками*. На рисунке 1 приведен пример конфигурации, причем источник помечен треугольником, а стоки — жирными точками.

Основная задача: для заданной конфигурации точек $K$ построить дерево $A$, корень которого совпадает с источником, множество концевых вершин которого содержит множество стоков конфигурации $K$ и задержка до всех концевых вершин дерева, являющихся стоками конфигурации, одинакова. Дерево $A$, построенное для конфигурации $K$ и обладающее данными свойствами, будем называть *синхронизирующим* и обозначать $A(K)$.

Неформально, синхронизирующее дерево позволяет доставлять сигнал от источника за одинаковое время до каждого из стоков. На рисунке 1 приведен пример синхронизирующего дерева, для определенной ранее конфигурации.

Отметим, что в синхронизирующем дереве могут быть концевые вершины не совпадающие со стоками.

*Расстоянием* между двумя точками плоскости $a = (x_a, y_a)$ и $a' = (x_{a'}, y_{a'})$ назовем число $\rho(a, a') = |x_a - x_{a'}| + |y_a - y_{a'}|$.

$r(K) = \min\limits_{a,a' \in K,\ a \neq a'} \rho(a, a')$ — минимальное расстояние между любыми двумя разными точками конфигурации $K$.

Для натурального $m$ введем следующий класс конфигураций

$$\mathcal{K}_m = \{K : r(K) \geq m\}.$$

Конфигурацию точек, для которой невозможно построить синхронизирующее дерево, будем называть *ловушкой*.

В работе [3] было показано, что в классах $\mathcal{K}_1$ и $\mathcal{K}_2$ существуют ловушки, и что для любой конфигурации из $\mathcal{K}_n$, где $n \geq 5$, существует синхронизирующее дерево. Оставался открытым вопрос для конфигураций из $\mathcal{K}_3$ и $\mathcal{K}_4$. Результаты данной работы позволяют получить окончательные ответы на данные вопросы.

**Теорема 1.** *В классе $\mathcal{K}_3$ существует ловушка.*

**Теорема 2.** *Для любой конфигурации $K \in \mathcal{K}_4$ можно построить синхронизирующее дерево.*

Идея доказательства теоремы 2 основывается на следующем алгоритме построения синхронизирующих деревьев, схематически изображенном на рисунке 2.

1) Соединяются по четыре соседних стока в одно синхронизирующее поддерево, с подкорнем в некоторой точке решетки. В случае если стоков оказывается меньше, то в нужном месте добавляем "фиктивное" ребро для сохранения симметрии.

2) Далее первый раз отражаем схему соединения относительно прямой на которой лежат 2 соседних соединенных стока поддерева, и второй раз относительно прямой перпендикулярной первой прямой. И соединяем отображенной схемой соседние стоки.

3) Соединяются два соседних подкорня, от которых были построены поддеревья доставляющие сигналы в предыдущем шаге одно поддерево. В случае если в каких-то поддеревьях не хватает стоков,

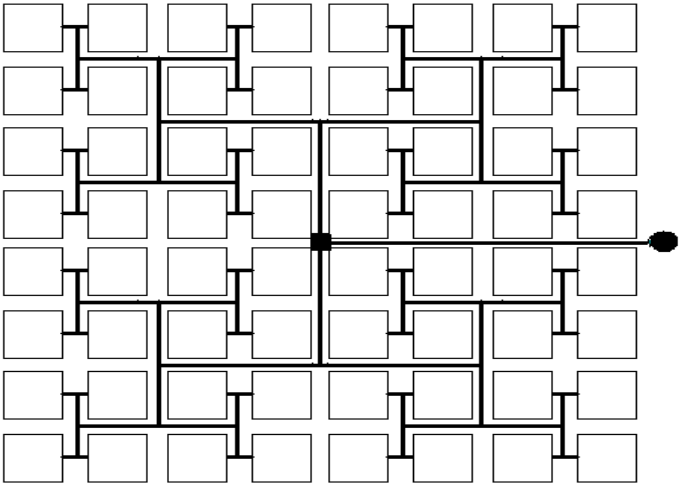


Рис. 2: Схема соединения поддеревьев

то в соответствующих местах добавляются для сохранения симметрии "фиктивные" ребра.

4) Повторяется 2-й шаг, до тех пор пока не будет уложено дерево для всех точек конфигурации.

Дерево, приведенное на рисунке 1, является деревом, построенным по данному алгоритму.

**Литература**

1. Pavisic I., Lu A., Zolotykh A.A., Gasanov E.E. Method in integrating clock tree synthesis and timing optimization for an integrated circuit design. United States Patent: 6,550,044, April 15, 2003

2. Sherwani N.A. Algorithms for VLSI Physical Design Avtomation. — Kluwer Academic Publishers, 1993.

3. Гасанов Э.Э., Проворова А.Л. О синтезе синхронизирующих деревьев // Материалы IX Международной конференции "Интеллектуальные системы и компьютерные науки"(23-27 октября 2006 г.), том 1, часть 1. - М.: Изд-во механико-математического факультета МГУ, 2006, с. 89-92.

# ПРЕДСТАВЛЕНИЯ ЭЛЕМЕНТОВ ЧАСТИЧНО-УПОРЯДОЧЕННЫХ АЛГЕБРАИЧЕСКИХ СИСТЕМ ФРАГМЕНТАМИ

**Грунский И.С., Максименко И.И.**
**(ИПММ НАН Украины, Донецк)**
*iim@bank-prsp.dn.ua*

**Введение**

Одной из важнейших проблем теории дискретных систем является анализ поведения объекта (автомата, помеченного графа) посредством проведения с ним экспериментов [1,2].

В работах [1,3] был введен и обоснован подход к исследованию контрольных и распознающих экспериментов в классах автоматов Мили на основе их представления специальными окрестностями в метрических пространствах автоматов. Для бэровской метрики, отражающей близость автоматов по поведению, были найдены [3] конструктивные критерии существования контрольных экспериментов.

Данный метод использован в [4] для неструктурированных объектов и их дескрипторов.

В настоящей статье этот подход распространен на алгебраические системы специального вида, обобщающие классы автоматов Мили и помеченных графов.

**1. Основные понятия и определения**

Рассмотрим алгебраическую систему вида $(\mathbf{A}, \leq, \mathrm{n})$, где $\mathbf{A}$ - счетное множество объектов, $\leq$ - предпорядок и $n : \mathbf{A} \to \mathbf{N}^+ \cup \{\infty\}$ - невозрастающая функция сложности.

Два объекта $A$ и $B$ эквивалентны ($A \cong B$), если одновременно $A \leq B$ и $B \leq A$. Каждый объект $A$ однозначно определяется множествами фрагментов $Fr(A) = \{B \in \mathbf{A} | \mathrm{B} \leq \mathrm{A}\}$ и кофрагментов $CoFr(A) = \{B \in \mathbf{A} | \mathrm{B} \not\leq \mathrm{A}\}$. Объект $A$ с конечной сложностью $n(A)$ назовем финитным и инфинитным в противном случае. Объект $C$ является разделяющим для объектов $A$ и $B$ ( $C \in S(A, B)$ ), если выполнено одно из условий ($C \leq A$ и $C \not\leq B$) или ($C \not\leq A$ и $C \leq B$).

Систему $(\mathbf{A}, \leq, \mathrm{n})$ назовем финитно разделимой, если для любых двух неэквивалентных объектов существует финитный объект, их разделяющий.

На множестве объектов введем "расстояние"между объектами $\beta$ аналогично "бэровской"метрике [3], полагая $\beta(A, B) = 0$, если $A \cong B$

и $\beta(A,B) = 1/r$, где $r = inf\{n(C)|C \in S(A,B)\}$ в противном случае. Через $LimF$ обозначим множество предельных объектов [4] для $F \subseteq \mathbf{A}$.

Пару объектов $(A,B) \in Fr(A_0) \times CoFr(A_0)$ назовем представлением для произвольных $A_0 \in \mathbf{A}$ и $F \subseteq \mathbf{A}$, если для любого $C \in F$ из включения $(A,B) \in Fr(C) \times CoFr(C)$ вытекает $C \cong A_0$.

Система $(\mathbf{A}, \leq, \mathrm{n})$ сильно непредставима, если для всякого объекта $A \in \mathbf{A}$ не существует представление относительно $A$ и $\mathbf{A}$.

Систему $(\mathbf{A}, \leq, \mathrm{n})$ назовем линейно упорядоченной, если $\mathbf{A}$ - линейно упорядочено.

Система $(\mathbf{A}, \leq, \mathrm{n})$ всюду плотна, если для любых объектов $A, B$ из соотношения $A < B$ вытекает существование объекта $C$, для которого $A < C < B$.

Введем алгебраические системы объектов вида $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$, где $\leq$ - предпорядок, $\nabla, \triangle$ - идемпотентные, коммутативные и ассоциативные всюду определенные бинарные операции, $\mathrm{n} : \mathbf{A} \to \mathbf{N}^+ \cup \{\infty\}$ - неубывающая функция сложности, и справедливы следующие аксиомы:

1. для любых двух объектов $A$ и $B$ выполнены соотношения $A \leq A\nabla B$ и $A\triangle B \leq A$;

2. для любых объектов $A_1, A_2, B \in \mathbf{A}$ из $A_1 \leq B, A_2 \leq B$ следует $A_1 \nabla A_2 \leq B$;

3. для любых объектов $A_1, A_2, B \in \mathbf{A}$ из $A_1 \not\leq B, A_2 \not\leq B$ вытекает $A_1 \triangle A_2 \not\leq B$;

4. для любых двух объектов $A, B \in \mathbf{A}$ полагаем, что $n(A\nabla B) = max(n(A), n(B))$ и $n(A\triangle B) = min(n(A), n(B))$.

Назовем алгебраическую систему $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ локально замкнутой, если для всякого $A \in \mathbf{A}$ множества $Fr(A)$ и $CoFr(A)$ замкнуты относительно счетного числа операций $\nabla$ и $\triangle$ соответственно.

Введем операцию $\odot$ над парами объектов $(A_1, B_1), (A_2, B_2) \in \mathbf{A^2}$, полагая $(A_1, B_1) \odot (A_2, B_2) = (A_1 \nabla A_2, B_1 \triangle B_2)$.

**2. Представимость алгебраических систем вида** $(\mathbf{A}, \leq, \mathrm{n})$

Имеет место следующий критерий сильной непредставимости:

**Утверждение 1** *Пусть дана линейно упорядоченная система* $(\mathbf{A}, \leq, \mathrm{n})$.

*Система сильно непредставима тогда и только тогда, когда она является всюду плотной.*

В работе [4] был получен метрический критерий существования финитных представлений неструктурированных объектов. Для произвольных алгебраических систем подобный критерий в общем случае не выполнен:

**Теорема 2** *Дана финитно разделимая алгебраическая система* $(\mathbf{A}, \leq, \mathrm{n})$.

*Если для произвольных* $A_0 \in \mathbf{A}$ *и* $F \subseteq \mathbf{A}$ *существует финитное представление, тогда* $A_0 \notin limF$. *Обратное утверждение неверно.*

**3. Представимость алгебраических систем вида** $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$

Алгебраическая структура систем вида $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ описана в

**Утверждение 3** *Алгебраическая система вида* $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ *является верхней полурешеткой, но не решеткой в общем случае.*

Справедливо

**Утверждение 4**

*Для локально замкнутых алгебраических систем* $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ *существует представление для всякого* $A_0 \in \mathbf{A}$ *и произвольного множества* $F \subseteq \mathbf{A}$.

Для финитно разделимых и локально замкнутых алгебраических систем вида $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ справедлив метрический критерий финитной представимости:

**Теорема 5**

*Финитное представление для* $A_0 \in \mathbf{A}$ *и множества* $F \subseteq \mathbf{A}$ *существует тогда и только тогда, когда* $A_0 \notin LimF$.

Алгебраическая структура представлений в финитно разделимых и локально замкнутых алгебраических системах $(\mathbf{A}, \leq, \nabla, \triangle, \mathrm{n})$ описывается следующей

**Теорема 6**

*1. Множество представлений для фиксированных* $A_0$ *и* $F$ *является идемпотентной и коммутативной полугруппой относительно операции* $\odot$.

*2. Множество финитных представлений для фиксированных* $A_0$ *и* $F$ *является идемпотентной и коммутативной полугруппой относительно операции* $\odot$.

**Заключение** В данной работе обобщен метрический критерий представимости, полученный ранее авторами для классов автоматов Мили[1,3] и классов неструктурированных объектов [4].

**Литература**

1. Грунский И. С., Козловский В. А. Синтез и идентификация автоматов. – Киев.: Наукова думка, 2004.

2. Бородай С. Ю. Эксперименты в эффективно заданных классах автоматов: Автореферат канд. физ.-мат. наук; 01.01.09 /СГУ - Саратов, 1997. - 21 с.

3. Максименко И. И. Эксперименты в финитно-определенных метрических пространствах автоматов: Автореферат канд. физ.-мат. наук; 01.01.09 /СГУ - Саратов, 2000. - 16 с.

4. Максименко И. И. Финитные представления неструктурированных объектов // Труды института прикладной математики и механики. - 2009 г. - том 19. -с. 162-167.

# РЕШЕТКА ЗАМКНУТЫХ КЛАССОВ САМОДВОЙСТВЕННЫХ ФУНКЦИЙ ТРЕХЗНАЧНОЙ ЛОГИКИ

**Жук Д.Н.**
**(МГУ имени М.В.Ломоносова)**
*zhuk@intsys.msu.ru*

В работе описывается решетка замкнутых классов трехзначной логики, которые вкладываются в предполный класс самодвойственных функций. Также в работе описаны различные свойства этой решетки: выделены все конечно-порожденные и предикато-описуемые классы; найдены мощности надрешеток и подрешеток для всех классов.

**Введение**

В работах [1,2] Э. Пост описал все замкнутый классы двузначной логики. Их оказалось счётное количество, причем все они конечно-порождены. Но в 1959 году было показано, что уже в трехзначной логике континуум замкнутых классов. С.В.Яблонский [4] описал все предполные классы трехзначной логики. Оказалось [5, 7], что во всех предполных классах кроме предполного класса линейных функций континуум замкнутых подклассов.

В настоящей работе исследуется структура замкнутых классов в предполном классе самодвойственных функций. Важные результаты в этой области были получены С. С. Марченковым. Он описал многие замкнутые классы [6], а также доказал, что решетка замкнутых классов самодвойственных функций континуальна [7]. Но несмотря на континуальность, автору удалось в явном виде описать структуру всех замкнутых классов самодвойственных функций. С помощью этого описания, в работе доказываются различные свойства полученной решетки. В частности выделяются все конечно-порожденные и предикатно-описуемые классы, найдены мощности подклассов и надклассов для каждого замкнутого класса.

**Описание решетки замкнутых классов**

Пусть $\mathbb{N} = \{1, 2, 3, \ldots\}$ — множество всех натуральных чисел, $\mathbb{N}_0 = N \cup \{0\}$, $E_k = \{0, 1, 2, \ldots, k-1\}$, $P_k$ — множество всех функций $k$-значной логики, $R_k$ — множество всех отношений или предикатов $k$-значной логики. В работе предикаты будем изображать в виде матриц, в которых столбцам соответствуют наборы, на которых предикат принимает значение 1. Для $S \subseteq R_k$ через $Pol(S)$ обозначим

множество всех функций из $P_k$, сохраняющих каждый предикат из множества $S$.

Пусть $m \in \mathbb{N}$, $n \in \mathbb{N}_0$, $A_1, \ldots, A_m \subseteq \{1, 2, \ldots, n\}$, $A_1 \cup \ldots \cup A_m = \{1, 2, \ldots, n\}$. (В случае, если $n = 0$ имеем $A_1 = A_2 = \ldots = A_m = \varnothing$). Тогда предикат $\pi_{A_1,\ldots,A_m} \in R^{m+n}$ определяется следующим соотношением:

$$\pi_{A_1,\ldots,A_m}(x_1, \ldots, x_m, y_1, \ldots, y_n) = 1$$

точно тогда, когда выполняются следующие условия:

1) $x_i \in \{0, 1\}$ для любого $i \in \{1, \ldots, m\}$;

2) $(x_i = 1) \vee (y_j \in \{0, 1\})$ для любых $i \in \{1, \ldots, m\}$, $j \in A_i$;

3) хотя бы одно из значений $x_1, \ldots, x_m, y_1, \ldots, y_n$ отлично от нуля.

Через $\Pi_n^m$ обозначим множество всех таких предикатов. Пусть $\Pi^l = \bigcup\limits_{3 \le m+n \le l} \Pi_n^m, \Pi_l = \bigcup\limits_{n \le l, m+n \ge 3} \Pi_n^m, \Pi = \bigcup\limits_l \Pi^l$.

На множестве $\Pi$ определяется рефлексивное и транзитивное бинарное отношение $\lesssim$ [9]. При этом доказывается, что для любого $\rho \in \Pi_n^m$ выполняется $\{\sigma \in \Pi | \sigma \lesssim \rho\} \subseteq \Pi^{m+n}$.

Пусть $\sigma : E_3 \longrightarrow E_3$, $\sigma(0) = 1$, $\sigma(1) = 0$, $\sigma(2) = 2$. Каждому предикату $\rho$ сопоставим предикат $\rho^*$, двойственный относительно замены нуля на единицу:

$$\rho^*(x_1, \ldots, x_n) := \rho(\sigma(x_1), \sigma(x_2), \ldots, \sigma(x_n)).$$

Для $S \subseteq R$ положим $S^* := \{\rho | \rho^* \in S\}$.

Будем говорить, что множество $F \subseteq \Pi$ замкнуто относительно отношения $\lesssim$, если

$$\forall \rho \in F, \forall \rho' \in \Pi \; (\rho' \lesssim \rho \implies \rho' \in F).$$

Пусть $F \subseteq \Pi$, положим

$$Clone(F) = Pol\left(F \cup \left\{\begin{pmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 2 \end{pmatrix}\right\}\right),$$

$$Clone^*(F) = Pol\left(F^* \cup \left\{\begin{pmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 2 \end{pmatrix}\right\}\right).$$

Теперь определим семейство замкнутых классов $\Upsilon$.

**Семейство $\Upsilon$.** Пусть $F \subseteq \Pi$ непусто и замкнуто относительно отношения $\lesssim$, тогда $Clone(F), Clone^*(F) \in \Upsilon$. Других замкнутых классов в семействе $\Upsilon$ нет.

**Теорема 1.** *Пусть* $F_1, F_2 \subseteq \Pi$ *непусты и замкнуты относительно отношения* $\lesssim$, *тогда* $Clone(F_1) \subseteq Clone(F_2) \Longleftrightarrow F_1 \supseteq F_2$.

**Следствие.** *Пусть* $F_1, F_2 \subseteq \Pi$ *непустые и замкнутые относительно отношения* $\lesssim$, *причём* $F_1 \neq F_2$, *тогда* $Clone(F_1) \neq Clone(F_2)$.

Определим некоторые предикаты, которые понадобятся нам в дальнейшем:

$$\rho_{+1} = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix}, \rho_T = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 2 \end{pmatrix}, \rho_N = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix},$$

$$\rho_W = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 2 \end{pmatrix}, \rho_Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix},$$

$$\rho_L(x_1, x_2, x_3) = 1 \Longleftrightarrow x_1 + x_2 = 2x_3 (mod\ 3),$$

$$\rho_{L2}(x_1, x_2, x_3, x_4) = 1 \Longleftrightarrow (\forall i\ x_i \in \{0,1\}) \wedge (x_1 + x_2 = x_3 + x_4 (mod\ 2)),$$

$$\rho_{\vee,n}(x_1, \ldots, x_n) = 1 \Longleftrightarrow$$
$$\Longleftrightarrow (\forall i\ x_i \in \{0,1\}) \wedge ((x_1 = 1) \vee (x_2 = 1) \vee \ldots \vee (x_n = 1)),$$

$$\rho_{\wedge,n}(x_1, \ldots, x_n) = 1 \Longleftrightarrow$$
$$\Longleftrightarrow (\forall i\ x_i \in \{0,1\}) \wedge ((x_1 = 0) \vee (x_2 = 0) \vee \ldots \vee (x_n = 0)),$$

$$\rho_{=,01}(x_1, x_2, x_3) = 1 \Longleftrightarrow$$
$$\Longleftrightarrow (x_1 = 1) \vee ((x_1 = 0) \wedge (x_2, x_3 \in \{0,1\}) \wedge (x_2 = x_3)),$$

$$\rho_{=,10}(x_1, x_2, x_3) = 1 \Longleftrightarrow$$
$$\Longleftrightarrow (x_1 = 0) \vee ((x_1 = 1) \wedge (x_2, x_3 \in \{0,1\}) \wedge (x_2 = x_3)),$$

$$\rho_{=,012}(x_1, x_2, x_3) = 1 \Longleftrightarrow (x_1 = 1) \vee ((x_1 = 0) \wedge (x_2 = x_3)),$$

$$\rho_{=,102}(x_1, x_2, x_3) = 1 \Longleftrightarrow (x_1 = 0) \vee ((x_1 = 1) \wedge (x_2 = x_3)),$$

Определим ещё два семейства замкнутых классов:

**Семейство Θ.**

$$\mathbf{S} = Pol\left(\{\rho_{+1}\}\right),\ \mathbf{S_0} = Pol\left(\{\rho_{+1}, (0)\}\right),\ \mathbf{SL} = Pol\left(\{\rho_{+1}, \rho_L\}\right),$$

$$\mathbf{1S} = [\{(x+1)(mod 3)\}],\ \mathbf{SL_0} = Pol\left(\{\rho_{+1}, \rho_L, (0)\}\right),$$

$$\mathbf{T} = Pol\left(\{\rho_{+1},\}\right),\ \mathbf{C} = Pol\left(\left\{\rho_{+1}, \begin{pmatrix}0 & 1\end{pmatrix}\right\}\right),$$

$$\mathbf{D} = Pol\left(\left\{\rho_{+1}, \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\right\}\right),\ \mathbf{M} = Pol\left(\left\{\rho_{+1}, \begin{pmatrix}0 & 0 & 1\\ 0 & 1 & 1\end{pmatrix}\right\}\right),$$

$$\mathbf{DM} = \mathbf{D} \cap \mathbf{M},\ \mathbf{DN} = Pol\left(\{\rho_{+1}, \rho_N, \rho_N^*\}\right),$$

$$\mathbf{TD} = \mathbf{T} \cap \mathbf{D},\ \mathbf{TM} = \mathbf{T} \cap \mathbf{M},\ \mathbf{TN} = \mathbf{DN} \cap \mathbf{T},$$

$$\mathbf{L} = Pol\left(\{\rho_{+1}, \rho_{L2}\}\right),\ \mathbf{TL} = \mathbf{L} \cap \mathbf{T},\ \mathbf{C_2} = \mathbf{L} \cap \mathbf{M},$$

$$\mathbf{TC_2} = \mathbf{C_2} \cap \mathbf{T},\ \mathbf{O} = [\{x\}].$$

**Семейство Φ.** Для $n \geq 2$

$$\mathbf{a_n} = Pol\left(\{\rho_{+1}, \rho_{\vee,n}\}\right),\ \mathbf{A_n} = Pol\left(\{\rho_{+1}, \rho_{\wedge,n}\}\right),$$

$$\mathbf{a_nM} = \mathbf{a_n} \cap \mathbf{M},\ \mathbf{A_nM} = \mathbf{A_n} \cap \mathbf{M},$$

$$\mathbf{a_nN} = Pol\left(\{\rho_{+1}, \rho_{\vee,n}, \rho_N\}\right),\ \mathbf{A_nN} = Pol\left(\{\rho_{+1}, \rho_{\wedge,n}\rho_N^*\}\right),$$

$$\mathbf{a}_\infty = \bigcap_n \mathbf{a_n},\ \mathbf{A}_\infty = \bigcap_n \mathbf{A_n},$$

$$\mathbf{a}_\infty\mathbf{M} = \bigcap_n \mathbf{a_nM},\ \mathbf{A}_\infty\mathbf{M} = \bigcap_n \mathbf{A_nM},$$

$$\mathbf{a}_\infty\mathbf{N} = \bigcap_n \mathbf{a_nN},\ \mathbf{A}_\infty\mathbf{N} = \bigcap_n \mathbf{A_nN},$$

$$\mathbf{aP} = Pol\left(\{\rho_{+1}, \rho_Q\}\right),\ \mathbf{AP} = Pol\left(\left\{\rho_{+1}, \rho_Q^*\right\}\right),$$

$$\mathbf{aPN} = Pol\left(\{\rho_{+1}, \rho_Q, \rho_N\}\right),\ \mathbf{APN} = Pol\left(\left\{\rho_{+1}, \rho_Q^*, \rho_N^*\right\}\right),$$

$$\mathbf{aP_1} = Pol\left(\{\rho_{+1}, \rho_Q, \rho_W,\}\right),\ \mathbf{AP_1} = Pol\left(\left\{\rho_{+1}, \rho_Q^*, \rho_W^*\right\}\right).$$

$\mathbf{aP_n} = \mathbf{aP_1} \cap Pol(\pi_{\{1,2,\ldots,n\}}), \mathbf{AP_n} = \mathbf{AP_1} \cap Pol(\pi^*_{\{1,2,\ldots,n\}})$, где $n \geq 2$.

$$\mathbf{aP}_\infty = \bigcap_n \mathbf{aP_n},\ \mathbf{AP}_\infty = \bigcap_n \mathbf{AP_n},$$

$$\mathbf{aQ} = Pol\left(\{\rho_{+1}, \rho_{=,01}\}\right),\ \mathbf{AQ} = Pol\left(\{\rho_{+1}, \rho_{=,10}\}\right),$$
$$\mathbf{aW} = Pol\left(\{\rho_{+1}, \rho_{=,012}\}\right),\ \mathbf{AW} = Pol\left(\{\rho_{+1}, \rho_{=,102}\}\right).$$

**Теорема 2.** *Множество* $\Upsilon \cup \Theta \cup \Phi$ *содержит все замкнутые классы, которые вкладываются в* $Pol\left(\{\rho_{+1}\}\right)$.

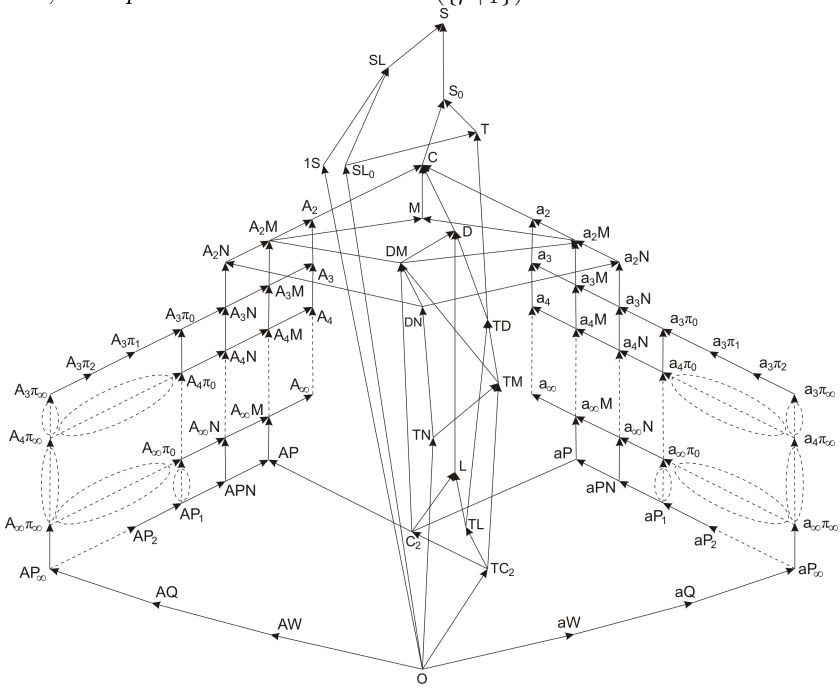



Рис. 1. Структура замкнутых классов.

Попарная вложимость замкнутых классов друг в друга для семейств $\Theta$ и $\Phi$ схематично изображена на рис. 1 в виде графа, где замкнутым классам соответствуют вершины графа. Две вершины графа $M_1$ и $M_2$ соединены сплошным ребром, причём $M_1$ расположена выше $M_2$, точно тогда, когда $M_2 \subset M_1$, и не существует замкнутого класса $M'$, такого что $M_2 \subset M' \subset M_1$. Две вершины графа $M_1$ и $M_2$ соединены пунктиром, причём $M_1$ расположена выше $M_2$, точно тогда, когда $M_2 \subset M_1$ и существует бесконечная последовательность замкнутых классов $K_1 \supset K_2 \supset K_3 \supset \ldots$, такая

что $K_1 \subset M_1$; $\bigcap\limits_i K_i = M_2$; если $M_2 \subset M' \subset M_1$, то $M' = K_i$ для какого-то $i$. В некоторых других случаях пунктирное ребро между двумя вершинами помещается в пунктирный эллипс. Это означает, что вложимость этих классов не подходит под предыдущие два случая.

Также на рис.1 изображены некоторые замкнутые классы из семейства $\Upsilon$. Для $n \geq 3$ положим

$$\mathbf{a_n}\pi_\mathbf{0} = Clone(\Pi^n \cap \Pi_0),\ \mathbf{A_n}\pi_\mathbf{0} = Clone^*(\Pi^n \cap \Pi_0),$$

$$\mathbf{a_n}\pi_\infty = Clone(\Pi^n),\ \mathbf{A_n}\pi_\infty = Clone^*(\Pi^n),$$

$$\mathbf{a_\infty}\pi_\mathbf{0} = Clone(\Pi_0),\ \mathbf{A_\infty}\pi_\mathbf{0} = Clone^*(\Pi_0),$$

$$\mathbf{a_\infty}\pi_\infty = Clone(\Pi),\ \mathbf{A_\infty}\pi_\infty = Clone^*(\Pi),$$

$$\mathbf{a_3}\pi_\mathbf{1} = Clone(\pi_{\{1\},\{1\}}),\ \mathbf{A_3}\pi_\mathbf{1} = Clone^*(\pi_{\{1\},\{1\}}),$$

$$\mathbf{a_3}\pi_\mathbf{2} = Clone(\pi_{\{1\},\varnothing}),\ \mathbf{A_3}\pi_\mathbf{2} = Clone^*(\pi_{\{1\},\varnothing}).$$

**Свойства семейств $\Theta$, $\Phi$ и $\Upsilon$**

Замкнутый класс $M \subseteq P_3$ называется конечно-порожденным, если существует конечное множество $M_0 \subseteq M$, такое что $M = [M_0]$. Замкнутый класс $M \subseteq P_3$ называется предикатно-описуемым, если существует конечное множество $S \subseteq R$, такое что $M = Pol(S)$. Как следует из предыдущей главы, все замкнутые классы в семействах $\Theta$ и $\Phi$ конечно-порождены. Положим

$$\rho_1 < \rho_2 \Longleftrightarrow \rho_1 \lesssim \rho_2 \wedge \neg(\rho_2 \lesssim \rho_1).$$

$$Bound(F) := \{\rho \in \Pi | \rho \notin F, \forall \sigma \in \Pi(\sigma < \rho \Longrightarrow \sigma \in F)\}.$$

**Теорема 3.** *Пусть $F \subseteq \Pi$, $F$ — непусто и замкнуто относительно отношения $\lesssim$, тогда $Clone(F)$ конечно-порожден точно тогда, когда множество $Bound(F)$ конечно.*

**Следствие.** *Пусть $F \subseteq \Pi$, $F$ — непусто и замкнуто относительно отношения $\lesssim$, $|F| < \infty$, тогда $Clone(F)$ конечно-порожден.*

**Теорема 4.** *Замкнутый класс $M \in \Theta \cup \Phi$ предикатно-описуем точно тогда, когда*

$$M \notin \{\mathbf{a_\infty}, \mathbf{A_\infty}, \mathbf{a_\infty M}, \mathbf{A_\infty M}, \mathbf{a_\infty N}, \mathbf{A_\infty N}, \mathbf{aP_\infty}, \mathbf{AP_\infty}\}.$$

**Теорема 5.** *Пусть $F \subseteq \Pi$, $F$ — непусто и замкнуто относительно отношения $\lesssim$, тогда $Clone(F)$ предикатно-описуем точно тогда, когда $F$ конечно.*

Пусть $M_1, M_2$ — замкнутые классы из $\Upsilon \cup \Phi \cup \Theta$, причём $M_1 \subset M_2$, тогда будем говорить, что $M_1$ — подкласс $M_2$, а $M_2$ — надкласс $M_1$. Следующие теоремы описывают мощность множества подклассов и надклассов для замкнутых классов из $\Upsilon \cup \Phi \cup \Theta$.

**Теорема 6.** *Пусть $F \subseteq \Pi$ — непусто и замкнуто относительно отношения $\lesssim$, тогда мощность множества подклассов $Clone(F)$ континуальна тогда и только тогда, когда $F \neq \Pi$. Мощность множества подклассов $\mathbf{A}_\infty\pi_\infty = Clone(\Pi)$ конечна.*

**Теорема 7.** *Пусть $M \in \Theta \cup \Phi$, тогда мощность множества подклассов $M$*

- *счётна, если $M \in \{\mathbf{aP}, \mathbf{aPN}, \mathbf{aP_1}, \mathbf{aP_2}, \mathbf{aP_3}, \ldots, \mathbf{AP}, \mathbf{APN}, \mathbf{AP_1}, \mathbf{AP_2}, \mathbf{AP_3}, \ldots\}$,*
- *континуальна, если $M \in \{\mathbf{S}, \mathbf{S_0}, \mathbf{C}, \mathbf{M}, \mathbf{a}_\infty, \mathbf{A}_\infty, \mathbf{a}_\infty\mathbf{M}, \mathbf{A}_\infty\mathbf{M}, \mathbf{a}_\infty\mathbf{N}, \mathbf{A}_\infty\mathbf{N}\}$, либо $M \in \bigcup\limits_{n\geq 2} \{\mathbf{a_n}, \mathbf{A_n}, \mathbf{a_nM}, \mathbf{A_nM}, \mathbf{a_nN}, \mathbf{A_nN}\}$,*
- *конечна в остальных случаях.*

**Теорема 8.** *Пусть $F \subseteq \Pi$ — непусто и замкнуто относительно отношения $\lesssim$, тогда мощность множества надклассов $Clone(F)$ континуальна, если $F$ содержит бесконечное подмножество, состоящее из попарно несравнимых предикатов; конечна, если $F$ — конечно; и счётна в остальных случаях.*

**Следствие.** *Для $n \geq 3$ мощность множества надклассов $\mathbf{a_n}\pi_\infty$ конечна.*

**Теорема 9.** *Пусть $M \in \Theta \cup \Phi$, тогда мощность множества надклассов $M$*

- *счётна, если $M \in \{\mathbf{a}_\infty, \mathbf{A}_\infty, \mathbf{a}_\infty\mathbf{M}, \mathbf{A}_\infty\mathbf{M}, \mathbf{a}_\infty\mathbf{N}, \mathbf{A}_\infty\mathbf{N}, \mathbf{aP}, \mathbf{aPN}, \mathbf{aP_1}, \mathbf{aP_2}, \mathbf{aP_3}, \ldots, \mathbf{AP}, \mathbf{APN}, \mathbf{AP_1}, \mathbf{AP_2}, \mathbf{AP_3}, \ldots\}$,*
- *континуальна, если $M \in \{\mathbf{aP}_\infty, \mathbf{AP}_\infty, \mathbf{aQ}, \mathbf{AQ}, \mathbf{aW}, \mathbf{AW}, \mathbf{C_2}, \mathbf{TC_2}, \mathbf{O}\}$,*
- *конечна в остальных случаях.*

Условно говоря, из этих теорем и следствия следует, что континуум замкнутых классов расположен на рис.1 вблизи точки $\mathbf{a}_\infty\pi_\infty$, так как для любого $n \geq 3$ мощность множества надклассов $\mathbf{a_n}\pi_\infty$ конечна и для любого $m \geq 1$ мощность множества надклассов $\mathbf{aP_m}$ счётна.

**Литература**


1. Post E. Determination of all closed systems of truth tables. Bull. Amer. Math. Soc. 26, 427, 1920.

2. Post E. Two-Valued Iterative Systems of Mathematical Logic. Princeton Univ. Press, Princeton, 1941.

3. Янов Ю. И., Мучник А. А. О существовании $k$-значных замкнутых классов, не имеющих конечного базиса // ДАН СССР. 1959. Т. 127, № 1. С. 44-46.

4. Яблонский С. В. О функциональной полноте в трехзначном исчислении. Докл. АН СССР 95, 1153-1155, 1954.

5. Demetrovics J., Hannak L.: The number of reducts of preprimial algebra. Algebra Universalis. Volume 16, Number 1, 178-185, 1983.

6. Марченков С. С., Деметрович Я, Ханнак Л. О замкнутых классах самодвойственных функций в $P_3$. Методы дискретного анализа и решении комбинаторных задач 34, 38-73, Москва, 1980.

7. Марченков С. С.. О замкнутых классах самодвойственных функций многозначной логики. Проблемы кибернетики 40, 261-266, 1983.

8. Бондарчук В. Г., Калужнин Л. А., Котов В. Н., Ромов Б. А. Теория Галуа для алгебр Поста I-II. Кибернетика 3, 1-10 (1969), 5, 1-9 (1969).

9. Жук Д. Н. Решетка замкнутых классов самодвойственных функций трехзначной логики. Издательство МГУ. 2011.

## О НЕКОТОРЫХ АСПЕКТАХ ТЕОРЕМЫ РИМАНА РОХА НА КОНЕЧНЫХ ГРАФАХ

**Иванов И.О. (Московский Государственный Университет имени М.В.Ломоносова)**
*truefet@gmail.com*

В своей работе [1] Matthew Baker и Serguei Norine спроецировали хорошо известную теорему Римана-Роха на графы, рассматривая на них целочисленные дивизоры. При этом они установили связь своей теории с несколько изменённой Chip-Firing Game, введённой Biggs'ом в [2]. Biggs в своих работах [3], [4], [5] проводит связь этой игры с хроматическим числом графа и полиномом Tutte, характеризующим степень связности графа. В [6] была доказана теорема Римана-Роха для дивизоров с рационально-значными элементами.

Данная работа посвящена необходимости нахождения эффективного алгоритма, находящего выигрышную стратегию, либо утверждающего, что её не существует. Этот же алгоритм поможет в нахождении размерности линейной оболочки дивизора, введённой в [1]. Поиску таких алгоритмов и посвящена эта работа.

**Теорема 1.** *Существует алгоритм, который решает Chip-Firing Game для дивизора на полном графе с $n$ вершинами за $2n^2+1$ операцию.*

**Теорема 2.** *Существует алгоритм, который решает Chip-Firing Game для дивизора на полном двудольном графе с $n$ и $m$ вершинами в долях соответственно за $2n^2+5n+3mn+m+1$ операций.*

### Благодарности

Благодарю моего научного руководителя Ирматова Анвара Адхамовича за ценные рекомендации и советы.

### Литература

1. Matthew Baker and Serguei Norine. Riemann-Roch and Abel-Jacobi theory on a finite graph.

2. N.Biggs. Algebraic potential theory of graphs.

3. N.Biggs. Chip-firing and the critical group of a graph.

4. N.Biggs. The Tutte polynomial as a growth function.

5. N. Biggs and P. Winkler. Chip-firing and the chromatic polynomial.

6. Rodney James and Rick Miranda. A Riemann-Roch theorem for edge-weighted graphs.

# О СЛОЖНОСТИ ТЕСТИРОВАНИЯ ЛОГИЧЕСКИХ УСТРОЙСТВ НА НЕКОТОРЫЕ ТИПЫ НЕИСПРАВНОСТЕЙ

**Икрамов А.А. (Московский Государственный Университет им. М.В. Ломоносова)**
*melan44@mail.ru*

В статье рассматриваются сложности тестирования на разнотипные неисправности. Для некоторых классов получены точные значения, для других верхние и нижние оценки. Также рассмотрен случай для почти всех булевых функций на класс инверсных неисправностей и нижняя оценка для перепутываний не более двух переменных

*Определение.* Неисправностью назовем отображение $\phi\colon E_2^n \to Q$, где $Q \subset E_2^n$ и $\exists \alpha \in E_2^n\colon \phi(\alpha) \neq \alpha$.

*Определение.* Проверяющим тестом для класса неисправностей $\Phi$ и функции $f \in P_2(n)$ назовем $T \subset E_2^n$ такое, что $\forall \phi \in \Phi \quad \exists\, \alpha \in T$:
$f(\phi(\alpha)) \neq f(\alpha)$.

*Определение.* Сложностью тестирования функции $f \in P_2(n)$ на класс неисправностей $\Phi$ назовем $L(f, \Phi) = \min |T|$ среди всех проверяющих тестов $T$.

*Определение.* Сложностью тестирования класса неисправностей $\Phi$ для $P_2(n)$ назовем величину $L(n, \Phi) = \max_{f \in P_2(n)} L(f, \Phi)$.

*Определение.* Классом инверсных неисправностей $F_{in}^2$ назовем множество всех $\phi_\sigma\colon \phi_\sigma(\alpha) = \alpha \oplus \sigma$, $\sigma \in E_2^n$
$\{0\}$. Через $F_{in}^2(p)$ обозначим класс таких $\phi_\sigma$, что $||\sigma|| = p$.

**Теорема 1.** *Для почти всех* $f \in P_2(n) \quad L(f, F_{in}^2(1)) = 1$.

*Доказательство.* Рассмотрим произвольный набор $\tilde{\alpha} \in E_2^n$. Все наборы $\tilde{\beta}$, находящиеся от него на расстоянии 1 по Хеммингу, образуют шар, в который рассматриваемая неисправность может перевести данный набор. Если $f(\tilde{\alpha}) \neq f(\tilde{\beta})$ для всех таких $\tilde{\beta}$, то набор $\tilde{\alpha}$ является тестовым для данной функции. Оценим количество функций, у которых существует такой набор. Число наборов, у которых нужно зафиксировать значения равно $n+1$. Теперь возьмем другие $n+1$ наборов, связанных требованием на расстояние. Чтобы генерируемые здесь функции не совпали, мы вычеркнем из возможных значений на предыдущих $n+1$ наборе то значение, что зафиксировали на предыдущем шаге. Получим $2^{n+1}-1$ значений на них. Таким образом, на каждом последующем шаге мы генерируем различные

функции. Посчитаем их общее количество:

$$\sum_{k=1}^{t(n)} 2^{2^n-k(n+1)} \cdot (2^{n+1}-1)^{k-1} = 2^{2^n-n-1} \sum_{k=0}^{t(n)-1} \left(\frac{2^{n+1}-1}{2^{n+1}}\right)^k \to_{n\to\infty}$$

$$\to 2^{2^n-n-1} \cdot \frac{1}{1-\frac{2^{n+1}-1}{2^{n+1}}} = 2^{2^n-n-1} \cdot 2^{n+1} = 2^{2^n}$$

Здесь $t(n)$ – число возможных разбиений булева куба на шары радиуса 1. Это число равно числу кодов Хемминга, исправляющих одну ошибку, то есть $t(n) = 2^{n-[\log_2 n]-1}$. Таким образом, почти все булевы функции имеют тестовый набор на $F_{in}^2(1)$.

*Определение.* Классом неисправностей типа конъюнктивных слипаний $S_{\&}^2$ назовем класс разбиений множества переменных $X^n$ и значением переменной $x_i$ будет являться минимальное значение из всех переменных ее множества.

*Определение.* Классом неисправностей типа конъюнктивных слипаний $S_{\vee}^2$ назовем класс разбиений множества переменных $X^n$ и значением переменной $x_i$ будет являться максимальное значение из всех переменных ее множества.

В [1] даются нижняя и верхняя оценки: $n-1 \le L(n, S_{\vee}^2 \cup F_{in}^2(1)) \le n$ (Предложение 18). Докажем следующее:

**Теорема 2.** $\forall n \ge 2 \quad L(n, S_{\vee}^2 \cup F_{in}^2(1)) = n$

*Доказательство.* Рассмотрим функцию $x_1 \& x_2 \& \ldots \& x_n$. Каждый набор слоя $n-1$ проверяет эту функцию на слипание переменной, равной на нем 0, с остальными переменными. Значит, достаточно $n-1$ набора, чтобы проверить на все неисправности типа слипания (оставшаяся переменная, которая среди всех взятых наборов принимает значение 1, уже проверена на слипание, так как все остальные от нее отделены). Однако, эта оставшаяся переменная не проверена на инверсию. Следовательно, необходимо добавить оставшийся набор из слоя $n-1$ для проверки на инверсию (либо набор из всех единиц). Таким образом, сложность тестирования этой функции равна $n$.

По принципу двойственности верно следующее утверждение (возьмем функцию $x_1 \vee x_2 \vee \ldots \vee x_n$):

**Теорема 3.** $\forall n \ge 2 \quad L(n, S_{\&}^2 \cup F_{in}^2(1)) = n$

Так как имеем $L(n, S^2_\vee) = n - 1$ (доказано в [1]), то по принципу двойственности верно $L(n, S^2_\vee) = n - 1$. Значит, очевидна оценка $L(n, S^2_\vee \cup S^2_\&) \leq 2(n-1)$. Докажем, что здесь верно равенство:

**Теорема 4.** $\forall n \geq 3 \quad L(n, S^2_\vee \cup S^2_\&) = 2(n-1)$

*Доказательство.* Рассмотрим функцию $(x_1 \vee x_2 \vee \ldots \vee x_n) \oplus x_1 \& x_2 \& \ldots \& x_n$. Для тестирования на все виды слипания нужно по $n-1$ набору со слоев 1 и $n-1$. Так как в этом случае они не пересекаются, то получаем равенство.

Из теорем 2, 3, 4 получаем следствие:

**Теорема 5.** $\forall n \geq 2 \quad 2n - 2 \leq L(n, S^2_\vee \cup S^2_\& \cup F^2_{in}(1)) \leq 2n - 1$

*Определение.* Перестановка $s$ множества индексов переменных из $X^n$ называется неисправностью типа перепутывания кратности $p = \sum_{i=1}^{n} \text{sign}|s(i) - i|$, если $p > 0$.

Обозначим класс неисправностей типа перепутывания над алгеброй логики через $W^2$. Класс таких неисправностей кратности $p$ через $W^2(p)$.

**Теорема 6.** $L(n, W^2(2)) \geq n - 1$

*Доказательство.* Построим следующую функцию: пусть она всюду равна 1, кроме следующих наборов: в слое $n-1$ набор $(1, \ldots, 1, 0)$, в слое $n-2$ набор $(1, \ldots, 1, 0, 0)$ и так далее до слоя 1 с набором $(1, 0, \ldots, 0)$. Тогда эти наборы будут тестовыми (их $n-1$): каждый проверяет уникальную неисправность (для набора из слоя $i$ перестановку переменных $x_i$ и $x_{i+1}$, которую никакие другие наборы не проверяют. Следовательно, сложность тестирования на перепутывание не более двух переменных не меньше $n-1$.

Выражаю благодарность за помощь в написании работы своему научному руководителю Кудрявцеву Валерию Борисовичу.

**Литература**

1. Кудрявцев В. Б., Гасанов Э. Э., Долотова О. А., Погосян Г. Р. Теория тестирования логических устройств. // — М.: Физматлит, 2006

## ДЕКОМПОЗИЦИЯ ГРАФА ПО СРЕДНЕЙ ПЛОТНОСТИ

**Магомедов А.М., Магомедов М.А. (Махачкала)**
*magomedtagir1@yandex.ru*

Для заданного графа $G = (V, E)$, где $V$ — множество вершин, $E$ — множество рёбер (допускаются параллельные рёбра и петли), обозначим наибольшую степень вершины через $\Delta(G)$, а наибольшее значение «средней плотности» подграфа $G' = (V', E') \subseteq G$ (отношения числа рёбер $|E'|$ к числу вершин $|V'|$) — через $mad\,(G)$.

В [1]–[2] построены алгоритмы вычисления $mad\,(G)$ за время $O(M(|V|, |V| + |E|) \log |V|)$, где $M(a, b)$ — время нахождения минимального разреза в сети с $a$ вершинами и $b$ рёбрами.

Понятие средней плотности графа используется в задачах оптимизации расписаний. Пусть каждому прибору из некоторого заданного множества предписан для обработки набор из двух требований — элементов множества $\{1, 2, \ldots, n\}$ (необязательно различных), над каждым из двух требований прибор должен выполнить операцию единичной длительности; одновременно требование (или прибор) может участвовать разве лишь в одной операции. Если $G = (V, E)$ — граф с множеством вершин $V = \{v_1, \ldots, v_n\}$, где количество рёбер вида $(v_i, v_j)$ в множестве $E$ равно количеству приборов с предписанным набором $(i, j)$, то результат статьи [3] формулируется следующим образом: *построение расписания нечетной длительности* $\Delta(G)$*, где каждый прибор выполняет запланированные операции в последовательные промежутки времени, возможно тогда и только тогда, когда*

$$mad\,(G) \leq \lfloor \Delta(G)/2 \rfloor . \tag{1}$$

Для $\Delta(G) = 5$ результат был получен ранее в [4].

Здесь и ниже ограничение рассмотрения графами нечетной степени объясняется не только соображениями краткости, но и тем обстоятельством, что рассмотрение для графов четной степени существенно проще (и базируется на классической теореме Петерсена о разбиении связного регулярного графа четной степени на гамильтоновы циклы [5]).

В задачах оптимизации расписаний (и интервальной рёберной раскраски графа) востребованы такие разбиения расписания (графа), где существование решения задачи для всего расписания (гра-

фа) равносильно существованию решения соответствующей подзадачи для каждого подрасписания (подграфа) — элемента разбиения.

Пусть $G = (V, E)$ — граф нечетной степени, $p_1$ и $p_2$ — нечетные натуральные числа,

$$p_1 + p_2 = \Delta(G) + 1; \quad 3 \le p_i \le \Delta(G) - 2, \quad i = 1, 2.$$

Разбиение множества $E$ на подмножества $E_1$ и $E_2$ будем называть *декомпозицией по средней плотности*, если подграфы $G_1$ и $G_2$ графа $G$, порожденные соответственно $E_1$ и $E_2$, удовлетворяют условиям:
1) для каждой вершины $v \in V$ равенства $d_{G_i} v = p_i$ и
$d_{G_{3-i}} v = p_{3-i} - 1$ равносильны
2) $\Delta(G_i) = p_i$
3) $mad\,(G_i) \le \lfloor p_i/2 \rfloor$ ; $i = 1, 2$

**Теорема.** *Декомпозиция графа $G$ по средней плотности существует тогда и только тогда, когда имеет место* (1).

Заметим, что свойство (1) при $\Delta(G) = 3$ является наследуемым и означает, что всякая связная компонента графа $G$ содержит не более одного цикла.

Работа выполнена при финансовой поддержке ФЦП «Научные и научно-педагогические кадры инновационной России» No 2011-1.3.2-111-017/12.

**Литература**

1. Picard J.-C., Queyranne M. A network flow solution to some nonlinear 0-1 programming problems, with applications to graph theory // Networks. – V. 12, 1982, p. 141-159.

2. Goldberg A. V. Finding a maximum density subgraph // Technical Report. – Berkeley / University of California, Computer Science Division, 1984 (Identifier: CSD-84-171).

3. Магомедов А. М. Непрерывное расписание с двухэлементными предписаниями // Известия Саратовского университета. Новая серия. Сер. «Математика. Механика. Информатика». – Т. 11, вып. 2, 2011, с. 113-119.

4. Сапоженко А. А., Магомедов А. М. Условия существования непрерывных расписаний длительности пять // Вестник МГУ, сер. «Вычислительная математика и кибернетика». – Т. 34, № 1, 2010, с. 39-44.

5. Petersen J. Die theorie der regularen graphen // Acta Math. – 15, 1891, p. 193–220. JBuch. 23.115.

# ИНТЕГРАЛЬНАЯ ФОРМУЛА ЧИСЛА ПОРОГОВЫХ ФУНКЦИЙ

**Носов М.В. (МГУ, мех. –мат.)**
*mvnosov@rambler.ru*

Пусть $F$ пороговая функция от $n$ переменных, $a = (a_0, a_1, \ldots, a_n), x = (x_1, \ldots, x_n)$. Известно, что её можно задать функцией $f_1(a, x)$, где

$$f_1(a, x) = a_1 x_1 + \ldots + a_n x_n + a_0,$$

$$a_0, a_1, \ldots, a_n \in \mathbf{Z},$$
$$|a_i| \leqslant P, i = 1, \ldots, n,$$
$$P = \left[(n+1)^{\frac{n+1}{2}} + 1\right].$$

Очевидно, тогда $F$ можно задать функцией

$$f(a, x) = 2a_1 x_1 + \ldots + 2a_n x_n + 2a_0 + 1,$$

при этом разделяющая гиперплоскость не проходит через вершины куба, т.е. принимает целые значения вне интервала $(-1, 1)$. Очевидно,что можно определить функцию $G(a, x)$, которая обеспечит "поразрядные"действия в нижеприведенных формулах. В выражении

$$\sum_{b,x} (f(a, x) f(b, x) G(b, x))$$

коэффициенты при $G(b, x)$ - целые положительные числа, если плоскости с направляющими векторами $a$ и $b$ определяют точку $x$ в полупространствах одного знака, в противном случае - целые отрицательные. По модулю коэффициенты не превосходят величины $(2(n+1)P)^2$.Если при фиксированном значении $b$ и всех векторах $x$, коэффициенты положительны, то это значит, что плоскости, определяемые векторами $a$ и $b$, делят вершины куба одинаково. Имеет место следующее очевидное утверждение. Если на конечном множестве $M$ введено отношение эквивалентности $\sim$, тогда

$$|M/\sim| = \sum_{x \in M} \frac{1}{\sum\limits_{y \in M} (x \sim y)}$$

Обозначим через $B$ множество всех целых точек куба $[-P,P]^{(n+1)}$

**Утверждение.** *Число пороговых функций задается формулой*

$$N_n = \int_0^1 \sum_{a\in B} \left( e^{2\pi i\left(\sum_{b\in B, x\in E_2^n}(f(a,x)f(b,x)G(b,x))\right)t} \right) \cdot$$

$$\cdot \left( \sum_{j=1}^{|B|} \frac{1}{j} \sum_{\substack{\beta\in B,\\ |\beta|=j}} \prod_{b\in\beta} \prod_{x\in E_2^n} \sum_{l=1}^{P} e^{-2\pi i l G(b,x)t} \cdot \right.$$

$$\left. \prod_{b\notin\in\beta} \left( \sum_{\gamma\subset E_2^n, atop|\gamma|\geq 1} \prod_{x\in\gamma} \left( \sum_{l=1}^{P} e^{-2\pi i l G(b,x)t} \right) \prod_{x\notin\gamma} \left( \sum_{l=1}^{P} e^{2\pi i l G(b,x)t} \right) \right) \right) dt$$

# ТЕОРЕТИКО-ВОЗМОЖНОСТНЫЕ МОДЕЛИ МАТРИЧНЫХ ИГР ДВУХ СУБЪЕКТОВ В ДВУХ ВАРИАНТАХ ТЕОРИИ ВОЗМОЖНОСТЕЙ

**Папилин С.С., Пытьев Ю.П. (Московский государственный университет имени М. В. Ломоносова)**
*papilin@physics.msu.ru, yuri.pytyev@gmail.com*

**1. Возможностная модель матричной игры двух субъектов** В игре участвуют два субъекта, «игрок A» и «игрок B». Игроки принимают нечеткие решения $\alpha \in \{1, \ldots, m\}$ и $\beta \in \{1, \ldots, n\}$ независимо друг от друга,

$$\mathrm{p}_i^{\mathrm{A}} \stackrel{\mathrm{def}}{=} \mathrm{P}^{\mathrm{A}}(\alpha = i) \geq 0, \max_{1 \leq i \leq m} \mathrm{p}_i^{\mathrm{A}} = 1, \tag{1}$$

$$\mathrm{p}_j^{\mathrm{B}} \stackrel{\mathrm{def}}{=} \mathrm{P}^{\mathrm{B}}(\beta = j) \geq 0, \max_{1 \leq j \leq n} \mathrm{p}_j^{\mathrm{B}} = 1, \tag{2}$$

есть распределения возможностей решений игроками. Наборы возможностей определяют нечеткие, или фазифицированные, стратегии принятия решений игроков A и B.

Определим матрицу переходных возможностей события $W$, элементы которой $s_{ij} \stackrel{\mathrm{def}}{=} \mathrm{P}(W|\alpha = i, \beta = j)$ задают зависимость переходной возможности $W$ от решений игроков, тогда *возможность события $W$ как функция нечетких стратегий игроков*

$$\mathrm{P}(W|\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) = \max_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} s_{ij} \bullet \mathrm{p}_i^{\mathrm{A}} \bullet \mathrm{p}_j^{\mathrm{B}} \stackrel{\mathrm{def}}{=} S(\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}), \tag{3}$$

где $\bullet$ есть операция умножения возможностей: минимум в первом варианте теории возможностей и «обычное» умножение во втором.

Будем считать, что в рассматриваемой игре для игрока A событие $W$ — «выигрыш», а для B — «проигрыш», и поэтому игрок A стремится максимизировать возможность 3, а игрок B — минимизировать.

*Максиминную стратегию* $\mathrm{p}^{*\mathrm{A}}$ *игрока* A определим как любое решение задачи

$$\min_{\mathrm{p}^{\mathrm{B}} \in \mathcal{P}^{\mathrm{B}}} S(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) = \max_{\mathrm{p}^{\mathrm{A}} \in \mathcal{P}^{\mathrm{A}}} \min_{\mathrm{p}^{\mathrm{B}} \in \mathcal{P}^{\mathrm{B}}} S(\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) \stackrel{\mathrm{def}}{=} s_{maxmin}, \tag{4}$$

где $s_{maxmin}$ назовем *максиминной возможностью*.

Множество всех максиминных стратегий по 4 задается условием

$$\min_{1\le j\le n}\max_{1\le i\le m} s_{ij}\bullet \mathrm{p}_i^{*\mathrm{A}} = \min_{1\le j\le n}\max_{1\le i\le m} s_{ij}. \qquad (5)$$

*Минимаксную стратегию* $\mathrm{p}_*^{\mathrm{B}}$ *игрока* B определим как любое решение задачи

$$\max_{\mathrm{p}^{\mathrm{A}}\in\mathcal{P}^{\mathrm{A}}} S(\mathrm{p}^{\mathrm{A}},\mathrm{p}_*^{\mathrm{B}}) = \min_{\mathrm{p}^{\mathrm{B}}\in\mathcal{P}^{\mathrm{B}}}\max_{\mathrm{p}^{\mathrm{A}}\in\mathcal{P}^{\mathrm{A}}} S(\mathrm{p}^{\mathrm{A}},\mathrm{p}^{\mathrm{B}}) \stackrel{\mathrm{def}}{=} s_{minmax}, \qquad (6)$$

в которой $s_{minmax}$ назовем *минимаксной возможностью.*

Множество всех минимаксных стратегий $\mathrm{p}_*^{\mathrm{B}}$ в 6 определяется условием

$$\max_{1\le j\le n}((\max_{1\le i\le m} s_{ij})\bullet \mathrm{p}_{*j}^{\mathrm{B}}) = \min_{1\le j\le n}\max_{1\le i\le m} s_{ij}. \qquad (7)$$

**Теорема 1.** *В любой одноматричной игре существуют максиминные* $\mathrm{p}^{*\mathrm{A}}$ *и минимаксные* $\mathrm{p}_*^{\mathrm{B}}$ *стратегии, причем максиминная возможность выигрыша и минимаксная возможность проигрыша равны*

$$s_{maxmin} = s_{minmax} \stackrel{\mathrm{def}}{=} s = \min_{1\le j\le n}\max_{1\le i\le m} s_{ij}. \qquad (8)$$

*Для любых максиминной* $\mathrm{p}^{*\mathrm{A}}$ *и минимаксной* $\mathrm{p}_*^{\mathrm{B}}$ *стратегий* $S(\mathrm{p}^{*\mathrm{A}},\mathrm{p}_*^{\mathrm{B}}) = s$*; тройка* $(\mathrm{p}^{*\mathrm{A}},\mathrm{p}_*^{\mathrm{B}},s)$ *есть решение одноматричной игры.*

*Четкая стратегия игрока A, в которой* $\mathrm{p}_i^{\mathrm{A}} = \begin{cases} 1, & i=i_0,\\ 0, & i\ne i_0,\end{cases}$ $i = 1,\dots,m$*, является максиминной, если* $\min_{1\le j\le n} s_{i_0 j} = s$.

*Четкие максиминные стратегии существуют не для любой матрицы переходных возможностей.*

*Четкая стратегия игрока B, в которой* $\mathrm{p}_j^{\mathrm{B}} = \begin{cases} 1, & j=j_0,\\ 0, & j\ne j_0,\end{cases}$ $j = 1,\dots,n$*, является минимаксной, если* $\max_{1\le i\le m} s_{ij_0} = s$.

*Четкие минимаксные стратегии существуют для любой матрицы переходных возможностей.*

В первом и втором вариантах теории возможностей формулировка теоремы выглядит одинаково, включая совпадение цен игры. Условия на множества всех максиминных и всех минимаксных стратегий отличаются операцией умножения возможностей.

**2. Возможностная модель биматричной игры**

В теории возможностей значения $P(W)$ и $P(\Omega \setminus W)$ не зависят друг от друга однозначно, и модель одноматричной игры не может охарактеризовать ситуацию, в которой игрок A считает «выигрышем» событие $W$, а игрок B считает «выигрышем» событие $\Omega \setminus W$. Для описания таких ситуаций следует ввести матрицы переходных возможностей $W$ и $\Omega \setminus W$:

$$s_{ij} \stackrel{\text{def}}{=} \mathrm{P}(W|\alpha = i, \beta = j),$$
$$t_{ij} \stackrel{\text{def}}{=} \mathrm{P}(\Omega \setminus W|\alpha = i, \beta = j),$$
$$\max(s_{ij}, t_{ij}) = 1, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n.$$

Соответственно

$$\mathrm{P}(W|\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) = \max_{\substack{1 \le i \le m \\ 1 \le j \le n}} s_{ij} \bullet \mathrm{p}_i^{\mathrm{A}} \bullet \mathrm{p}_j^{\mathrm{B}} \stackrel{\text{def}}{=} S(\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}), \tag{9}$$

$$\mathrm{P}(\Omega \setminus W|\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) = \max_{\substack{1 \le i \le m \\ 1 \le j \le n}} t_{ij} \bullet \mathrm{p}_i^{\mathrm{A}} \bullet \mathrm{p}_j^{\mathrm{B}} \stackrel{\text{def}}{=} T(\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}}). \tag{10}$$

Для рассматриваемой игры можно поставить две задачи: максимизации и минимизации.

В задаче максимизации цель игрока A — максимизировать $\mathrm{P}(W|\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}})$, а цель игрока B — максимизировать $\mathrm{P}(\Omega \setminus W|\mathrm{p}^{\mathrm{A}}, \mathrm{p}^{\mathrm{B}})$. *Точка равновесия* в такой задаче — пара стратегий $(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}})$, для которых выполняется условие

$$\forall \mathrm{p}^{\mathrm{A}} \in \mathcal{P}^{\mathrm{A}} \quad S(\mathrm{p}^{\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}}) \le S(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}});$$
$$\forall \mathrm{p}^{\mathrm{B}} \in \mathcal{P}^{\mathrm{B}} \quad T(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) \le T(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}}).$$

В задаче минимизации цель игрока A — минимизировать $\mathrm{P}(\Omega \setminus W)$, а цель игрока B — минимизировать $\mathrm{P}(W)$. *Точка равновесия* — пара стратегий $(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}})$, для которых

$$\forall \mathrm{p}^{\mathrm{A}} \in \mathcal{P}^{\mathrm{A}} \quad S(\mathrm{p}^{\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}}) \ge S(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}});$$
$$\forall \mathrm{p}^{\mathrm{B}} \in \mathcal{P}^{\mathrm{B}} \quad T(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}^{\mathrm{B}}) \ge T(\mathrm{p}^{*\mathrm{A}}, \mathrm{p}_*^{\mathrm{B}}).$$

**Теорема 2.** *В любой биматричной игре с задачей максимизации существуют точки равновесия. Пара четких стратегий* $(i^*, j_*)$

*есть точка равновесия тогда и только тогда, когда*

$$\max_{1\le i\le m} s_{ij_*} = s_{i^*j_*}; \quad \max_{1\le j\le n} t_{i^*j} = t_{i^*j_*}.$$

*Точки равновесия из четких стратегий могут как существовать, так и не существовать в зависимости от матриц* $\{s_{ij}\}$ *и* $\{t_{ij}\}$ *переходных возможностей.*

*В биматричной игре с задачей минимизации точки равновесия могут как существовать, так и не существовать в зависимости от матриц переходных возможностей. Если точки равновесия существуют, то среди них есть и точки равновесия из четких стратегий. Пара четких стратегий* $(i^*, j_*)$ *есть точка равновесия тогда и только тогда, когда*

$$\min_{1\le j\le n} s_{i^*j} = s_{i^*j_*}; \quad \min_{1\le i\le m} t_{ij_*} = t_{i^*j_*}.$$

*Если таких пар нет, то в соответствии с вышесказанным точек равновесия, в том числе из фазифицированных стратегий, в задаче минимизации нет.*

В первом и втором вариантах теории возможностей формулировка теоремы выглядит одинаково.

Работа выполнена при финансовой поддержке РФФИ, проект №11-07-00722-а.

**Литература**

1. Папилин С. С, Пытьев Ю. П. Вероятностные и возможностные модели матричных игр двух субъектов // Математическое моделирование. — 2010, т. 22, №12, с. 10–15.

2. Пытьев Ю. П. Возможность как альтернатива вероятности. Математические и эмпирические основы, применение. — М.: Физматлит, 2007.

# О ЧАСТОТНЫХ ЯЗЫКАХ НА БИГРАММАХ

**Петюшко А.А. (Московский Государственный Университет им. М. В. Ломоносова)**
*petsan@newmail.ru*

Пусть $A$ ($|A| < \infty$) - конечный алфавит, а $L \subseteq A^*$ - некоторый язык над этим алфавитом.

По каждому слову $\alpha$ языка $L$ можно построить матрицу биграмм $(n(\alpha))_{a,b\in A}$, такую что $n_{ab}(\alpha)$ - это число рядом рядом стоящих букв $ab$ в слове $\alpha$. В данной статье решается обратная задача - по матрице $n(\alpha)$ установить некоторые свойства языка $L(n(\alpha))$, то есть множества всех слов, имеющих матрицу биграмм $n(\alpha)$. Полученные языки $L(n(\alpha))$ удается классифицировать.

**Пример.** Пусть $A = \{0, 1\}$, $\alpha = 01011100$.

Тогда матрица биграмм $n(\alpha) = \begin{pmatrix} n_{00}(\alpha) & n_{01}(\alpha) \\ n_{10}(\alpha) & n_{11}(\alpha) \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 2 \end{pmatrix}$.

Рассмотрим сначала результат, касающийся регулярности языка, в котором заданы некоторые ограничения на какое-то подмножество элементов матрицы биграмм.

**Теорема 1.** *Пусть задан набор $k < \infty$ биграмм $\overline{\beta} = (\beta_1, ..., \beta_k)$, где $|\beta_i| = 2, i = 1..k$, а также набор отрезков $\overline{c} = ([c_1^1, c_2^1], ..., [c_1^k, c_2^k])$, где $c_1^i \leq c_2^i, c_j^i \in N \cup \{0\}, i = 1..k, j = 1..2$. Тогда язык $L_{\overline{\beta},\overline{c}} = \{\alpha | \quad n_{\beta_i}(\alpha) \in [c_1^i, c_2^i], i = 1..k\}$ регулярен.*

Более интересный случай, когда мы рассматриваем матрицу биграмм не как абсолютное ограничение, а как задание относительных значений биграмм, то есть языка, в котором сохраняются отношения $n_{ab}(\alpha)/n_{cd}(\alpha) \quad \forall a, b, c, d \in A, n_{cd}(\alpha) > 0$. Для более детального рассмотрения нам потребуется ряд определений.

**Определение.** *Назовем частотным языком на биграммах, заданным матрицей биграмм $n(\alpha)$, следующий язык при $k \in N$:*

$$F_{\cup n(\alpha)} = \bigcup_{k=1}^{\infty} L(kn(\alpha)).$$

Построим по матрице $n(\alpha)$ ориентированный граф $G_{n(\alpha)}$ на плоскости. Вершинами у этого графа будут все буквы из алфавита $A$, при этом ребра будут соответствовать биграммам с учетом их кратностей, то есть кратность $n_{ab}(\alpha)$ будет порождать $n_{ab}(\alpha)$ ориентиро-

ванных ребер $a \to b$. Аналогично, кратность $n_{cc}(\alpha)$ будет порождать $n_{cc}(\alpha)$ петель $c \to c$.

**Определение.** *Назовем ориентированный граф эйлеровым, если если выполняются следующие условия: 1) Все вершины, являющиеся начальной или конечной вершиной хотя бы одного ребра, лежат в одной компоненте связности соответствующего неориентированного графа; 2) У всех вершин количество входящих ребер равно количеству исходящих ребер.*

**Определение.** *Назовем ориентированный граф почти эйлеровым, если выполняются следующие условия: 1) Все вершины, являющиеся начальной или конечной вершиной хотя бы одного ребра, лежат в одной компоненте связности соответствующего неориентированного графа; 2) У всех вершин, кроме двух, количество входящих ребер равно количеству исходящих ребер. У оставшихся двух вершин разность количества входящих ребер и количества исходящих ребер равна +1 и -1 соответственно.*

Как показано в [1], в эйлеровом графе существует эйлеров цикл (то есть такой цикл, который содержит все ребра, причем каждое - только один раз), а в почти эйлеровом - эйлеров путь, не являющийся эйлеровым циклом (то есть такой путь, который содержит все ребра, причем каждое - только один раз, и при этом начальная вершина не совпадает с конечной).

**Теорема 2.** *Пусть задана матрица биграмм $n(\alpha)$. Тогда:*
*1) Если ориентированный граф $G_{n(\alpha)}$ является эйлеровым, то в частотном языке $F_{\cup n(\alpha)}$ счетное число слов;*
*2) Если ориентированный граф $G_{n(\alpha)}$ является почти эйлеровым, то в частотном языке $F_{\cup n(\alpha)}$ конечное ненулевое число слов, имеющих одинаковую длину;*
*3) Если ориентированный граф $G_{n(\alpha)}$ не является ни эйлеровым, ни почти эйлеровым, то в частотном языке $F_{\cup n(\alpha)}$ нет ни одного слова.*

Очевидно, что если выполняются условия 2) или 3) Теоремы 2, то язык $F_{\cup n(\alpha)}$, в котором не более чем конечное число слов, будет регулярным. Поэтому интересен вопрос, когда он будет являться регулярным при условии 1).

**Определение.** *Назовем две ненулевые матрицы $n_1$ и $n_2$ одинакового размера неколлинеарными, если не существует ненулевых действительных коэффициентов $c_1, c_2 \in R, (c_1, c_2) \neq (0, 0)$, таких,*

*что верно* $c_1 n_1 + c_2 n_2 = 0$.

**Теорема 3.** *Пусть* $A, |A| < \infty$ *- некоторый конечный алфавит. Далее, пусть задана матрица биграмм* $n(\alpha)$ *такая, что соответствующий ей ориентированный граф* $G_{n(\alpha)}$ *является эйлеровым. Тогда:*
*1) Если существует такое разложение* $n(\alpha)$ *в сумму двух ненулевых неколлинеарных матриц* $n(\alpha) = n(\alpha_1) + n(\alpha_2)$ *такое, что обе матрицы* $n(\alpha_1)$ *и* $n(\alpha_2)$ *задают ориентированные графы* $G_{n(\alpha_1)}$ *и* $G_{n(\alpha_2)}$*, которые являются эйлеровыми, то язык* $F_{\cup n(\alpha)}$ *нерегулярен;*
*2) В противном случае язык* $F_{\cup n(\alpha)}$ *регулярен.*

Однако данная теорема дает слишком общие условия на матрицу биграмм. Рассмотрим частный, но часто используемый на практике случай двухбуквенного алфавита.

**Теорема 4.** *Пусть* $A = \{0, 1\}$*. Далее, пусть задана матрица биграмм* $n(\alpha)$ *такая, что соответствующий ей ориентированный граф* $G_{n(\alpha)}$ *является эйлеровым. Тогда:*
*1) Язык* $F_{\cup n(\alpha)}$ *нерегулярен, если* $\exists i, i \in \{0, 1\}$ *такое, что* $n_{ii}(\alpha) > 0$*, и при этом* $\exists u \neq v, u, v \in \{0, 1\}$ *такие, что* $n_{uv}(\alpha) > 0$*;*
*2) Язык* $F_{\cup n(\alpha)}$ *регулярен, если* $\exists i, i \in \{0, 1\}$ *такое, что* $n_{ii}(\alpha) > 0$*, и при этом* $\forall u, v \in \{0, 1\}, (i, i) \neq (u, v)$ *выполняется* $n_{uv}(\alpha) = 0$*;*
*3) Язык* $F_{\cup n(\alpha)}$ *регулярен при* $n_{00}(\alpha) = n_{11}(\alpha) = 0$.

Отметим, что для доказательства двух последних теорем напрямую использовалась теорема Клини о представимости регулярных событий в автомате (см. [2]).

Автор выражает благодарность своему научному руководителю, д. ф.-м. н., профессору Бабину Д. Н., за постановку задачи и ценные указания.

**Литература**

1. Оре О. Теория графов. – М.: Наука, 1980.

2. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов. – М.: Наука, 1985.

# О ДВУХ МЕТОДАХ РАСПОЗНАВАНИЯ ЭКВИВАЛЕНТНОСТИ В АЛГЕБРАИЧЕСКИХ МОДЕЛЯХ ПРОГРАММ

**Подловченко Р.И. (НИВЦ МГУ им. М.В. Ломоносова)**
**Захаров В.А. (ВМК МГУ им. М.В. Ломоносова)**
*rip@vvv.srcc.msu.su, zakh@cs.msu.su*

Назначение данной статьи — обратить внимание на один из последних результатов в теории моделей последовательных программ. В ней даётся представление об алгебраических моделях программ, об основных проблемах их теории, условиях, в которых они рассматриваются, и концепциях, лежащих в основе двух практикуемых методов распознавания эквивалентности. Формулируются результаты, полученные этими методами и применимые в программировании.

Алгебраические модели программ введены в [4] как обобщение двух моделей последовательных программ — операторных схем Ляпунова-Янова [3,6] и дискретных преобразователей Глушкова-Летичевского [1]. Объекты алгебраической модели именуются схемами программ. В теории алгебраических моделей программ основной является проблема эквивалентных преобразований (э.п.) в модели. Она заключается в построении системы э.п., полной в модели, т.е. удовлетворяющей требованию: для любых двух эквивалентных схем из этой модели существует конечная цепочка э.п., принадлежащих системе и транслирующая одну из схем в другую. Проблема э.п. рассматривается только в моделях с разрешимой проблемой эквивалентности, т. е. предполагается наличие алгоритма, который распознает эквивалентность схем в модели. Таким образом, на первое место в исследованиях выходит проблема эквивалентности в модели.

Алгебраические модели строятся над двумя конечными алфавитами — алфавитом $Y$ операторных символов и алфавитом $P$ логических переменных, принимающих значения 0 и 1. Все алгебраические модели программ имеют общим множество своих объектов — схем программ и отличаются друг от друга отношением эквивалентности между схемами. Эти отношения вводятся единообразно, определяя тем самым всё множество моделей. Существенной является

**Теорема 1.** *Проблемы эквивалентности и э.п. в любой алгебраической модели программ над $Y, P$ сводятся к одноимённым проблемам в моделях матричных схем над $Y$, $P$.*

На основании этой теоремы обе проблемы рассматриваются в моделях матричных схем. Опишем их. Матричная схема над $Y, P$ представляет собой конечный граф с двумя выделенными вершинами — входом без заходящих в него дуг и выходом без исходящих из него дуг; остальные вершины наделены метками из $Y$; из каждой вершины графа, кроме выхода, исходят дуги в количестве, равном числу наборов значений всех переменных из $P$ (множество таких наборов обозначается $X$), и помечены различными наборами.

Матричные схемы выполняются на функциях, отображающих множество всех цепочек операторных символов из $Y$ (они называются операторными цепочками) в множество $X$. Такие функции называются функциями разметки. Выполнение схемы на функции разметки заключается в обходе схемы, который начинается в ее входе с пустой операторной цепочкой и сопровождается приписыванием к текущей операторной цепочке символа, сопоставленного вершине при переходе через нее; при этом выход из вершины происходит по дуге, помеченной тем набором из $X$, который является значением функции разметки на полученной цепочке. Результатом выполнения схемы считается цепочка, полученная к моменту достижения выхода схемы; иначе результат не определен. Эквивалентность схем определяется выбором двух параметров: отношения эквивалентности $\nu$ в множестве операторных цепочек над $Y$ и подмножества $L$ допустимых функций разметки: требуется, чтобы на любой допустимой функции разметки результат выполнения на ней одной из схем был определён, если определён результат выполнения другой, и эти результаты были $\nu$-эквивалентными. С введением такой эквивалентности получается $(\nu, L)$-модель матричных схем над $Y, P$.

К настоящему времени предложены два метода разрешения эквивалентности в моделях матричных схем — один в [5] и другой в [2]. Оба метода рассматривают сочетаемые маршруты в схемах, сравниваемых на эквивалентность, т. е пути в схемах, начинающиеся в их входах и пролагаемые общей для них допустимой функцией разметки. В терминах сочетаемых маршрутов формулируется критерий эквивалентности схем. Выделяется семейство упорядоченных моделей: в такой модели любая операторная цепочка не имеет $\nu$-эквивалентных ей подцепочек, а допустимыми являются все функции разметки, сохраняющие своё значение на $\nu$-эквивалентных операторных цепочках. Основанием этому является то, что в таких мо-

делях сочетаемость маршрутов алгоритмически распознаваема, а в матричной схеме любой маршрут прокладывается некоторой допустимой функцией разметки.

Метод в [5] нацелен на распознавание эквивалентности, в процессе которого устанавливаются свойства структуры эквивалентных схем. Последнее необходимо для решения проблемы э.п. Концепция этого метода — осуществить проверку эквивалентности двух схем просмотром сочетаемых маршрутов в них, имеющих конечную длину, которая определяется размерами схем. Доказана

**Теорема 2.** *Проблема эквивалентности в уравновешенной модели с левым и правым сокращением разрешима за полиномиальное время относительно размеров сравниваемых схем; в такой модели решена и проблема э.п.*

Названные в теореме 2 модели — это частный вид упорядоченных моделей, в которых $\nu$-эквивалентные цепочки равны по длине и сохраняют $\nu$-эквивалентность при сокращении их на $\nu$-эквивалентные префиксы (левая сократимость) и суффиксы (правая сократимость).

Альтернативный подход к построению эффективных алгоритмов решения задачи проверки эквивалентности схем программ предусматривает сведение этой задачи к проблеме пустоты для двухленточных односторонних детерминированных машин (2-DM) специального вида. Для проверки эквивалентности двух матричных схем $\pi_1, \pi_2$ в $(\nu, L)$-модели необходимо построить 2-DM $D_\nu$, описывающую отношение эквивалентности $\nu$ операторных цепочек, т. е. распознающую все $\nu$-эквивалентные пары операторных цепочек. Доказана

**Теорема 3.** *Отношение эквивалентности $\nu$ операторных цепочек может быть описано 2-DM тогда и только тогда, когда $(\nu, L)$-модель является упорядоченной.*

Далее для пары схем $\pi_1, \pi_2$ и 2-DM $D_\nu$, описывающей отношение эквивалентности $\nu$ операторных цепочек, строится комбинированная 2-DM $K(\pi_1, \pi_2, D_\nu)$, которая принимает в качестве входных данных, записанных на ее лентах, пары маршрутов в схемах $\pi_1$ и $\pi_2$. Устройство комбинированной 2-DM таково, что справедлива

**Теорема 4.** *Если 2-DM $D_\nu$ описывает эквивалентность $\nu$ операторных цепочек, то схемы $\pi_1$ и $\pi_2$ эквивалентны в $(\nu, L)$-модели тогда и только тогда, когда комбинированная машина $K(\pi_1, \pi_2, D_\nu)$*

*A: распознает пустое бинарное отношение и*
*B: в каждом бесконечном прогоне бесконечно часто считывает данные на обеих лентах.*

Разрешимость проблемы пустоты для некоторых классов комбинированных машин $K(\pi_1, \pi_2, D_\nu)$ обеспечивает

**Теорема 5.** *Если эквивалентность $\nu$ операторных цепочек описывается 2-DM $D_\nu$, имеющей конечное множество $F$ допускающих состояний, и схемы $\pi_1$ и $\pi_2$ эквивалентны в $(\nu, L)$-модели, то число состояний комбинированной машины $K(\pi_1, \pi_2, D_\nu)$, достижимых из ее начального состояния, ограничено величиной $2^{O(|F|(|\pi_1|+|\pi_2|))}$.*

**Следствие 1.** *Если отношение эквивалентности $\nu$ операторных цепочек разрешимо за полиномиальное время и описывается 2-DM $D_\nu$, имеющей конечное множество $F$ допускающих состояний, то проблема эквивалентности схем программ в $(\nu, L)$-модели принадлежит классу сложности co-NP.*

**Следствие 2.** *Если отношение эквивалентности $\nu$ операторных цепочек описывается конечной 2-DM $D_\nu$, то проблема эквивалентности схем программ в $(\nu, L)$-модели принадлежит классу сложности NLOG.*

Работа выполнена при поддержке ФЦП "Научные и научно-педагогические кадры инновационной России" 2009-2013 г.г.

**Литература**

1. Глушков В. М., Летичевский А. А. Теория дискретных преобразователей // Избранные вопросы алгебры и логики. – Новосибирск: Наука, 1973. – с. 5-39.

2. Захаров В. А. Проверка эквивалентности программ при помощи двухленточных автоматов // Кибернетика и системный анализ. – 2010. - N 4. – с. 39-48.

3. Ляпунов А. А. О логических схемах программ // Проблемы кибернетики. Вып.1. – М: Физматгиз. 1958. – с. 46-74.

4. Подловченко Р. И. Иерархия моделей программ // Программирование. – 1981. – N 2. – с. 3-14.

5. Подловченко Р. И. Об одной методике распознавания эквивалентности в алгебраических моделях программ // Программирование. – 2011. – N 6.

6. Янов Ю.И. О логических схемах алгоритмов // Проблемы кибернетики. Вып.1. – М: Физматгиз. 1958. – с. 75-127.

# О ПРОВЕРКЕ ЭКВИВАЛЕНТНОСТИ ПОСЛЕДОВАТЕЛЬНЫХ И РЕКУРСИВНЫХ ПРОГРАММ НА УПОРЯДОЧЕННЫХ ПОЛУГРУППОВЫХ ШКАЛАХ

**Подымов В.В. (ВМК МГУ им. М.В. Ломоносова)**
*valdus@yandex.ru*

Проблема эквивалентности программ в широком смысле формулируется следующим образом: требуется выяснить, имеют ли заданные программы одинаковое поведение. В данной работе в качестве формализации понятия программы используется модель рекурсивных программ, предложенная в заметке [1] и обобщающая модель вычислений последовательных программ.

В работе [2] была предложена методика исследования проблемы эквивалентности линейных унарных рекурсивных программ, семантика которых описывается уравновешенными полугрупповыми шкалами. Цель данного исследования состоит в обобщении этой методики на более широкий класс семантик, описываемых упорядоченными полугрупповыми шкалами.

Считаем заданными конечный алфавит $A$ базовых функций и конечный алфавит $C$ базовых предикатов. Также задан счетно-бесконечный алфавит $F$ заголовков функций, которые могут использоваться в программе. Слово в алфавите $A \cup F$ будем называть термом. Множество всех термов будем обозначать записью $Term$. Терм $t$ будем называть базовым, если $t \in A^*$, и линейным, если он является базовым или представим в виде $t = t'ft''$, где $t'$, $t''$ — базовые термы, $f \in F$.

Под унарной рекурсивной программой будем понимать систему $\pi = (F_\pi, D, T)$, где $F_\pi \subset F$ — конечное множество заголовков функций, определяемых в программе, $D : F_\pi \times C \to Term$ — описание функций, $T \in Term$ — запрос программы. Унарную рекурсивную программу будем называть линейной, если ее запрос и область значений функции $D$ суть линейные термы. Линейную унарную рекурсивную программу далее будем называть просто программой.

Сложность $|\pi|$ программы $\pi = (F_\pi, D, T)$ определим следующим образом: $|\pi| = |T| + \sum_{f \in F_\pi, c \in C} |D(f, c)|$.

Семантика программы определяется моделью — системой $M = (S, s_0, R, \xi)$, где $S$ — произвольное множество состояний данных, $s_0 \in S$ — начальное состояние, $R : S \times A \to S$ — функция преобразования данных, $\xi : S \to C$ — оценка истинности предикатов

на состояниях данных. Наряду с функцией $R$ будем использовать ее расширение $R^*$ с множества $A$ на множество базовых термов: $R^*(s, \lambda) = s$, $R^*(s, ah) = R^*(R(s, a), h)$. Вместо записи $R^*(s_0, h)$ для краткости будем использовать запись $[h]$.

Трассой программы $\pi = (T, F, D)$ будем называть конечную или бесконечную последовательность термов, начинающуюся с запроса программы и такую, что каждый следующий терм получается из предыдущего заменой входящего в него заголовка функции $f$ на терм $D(f, c)$, где $c$ — произвольный базовый предикат. Трассу программы $\pi$ будем называть ее вычислением в модели $M = (S, s_0, R, \xi)$, если выполнены следующие условия:

1. если она конечна, то оканчивается базовым термом и
2. при замене терма $t'ft''$, $f \in F$, на терм $t'D(f, c)t''$ базовый предикат $c$ определяется по правилу $c = \xi([t'])$.

В заданной модели $M$ у заданной программы $\pi$ существует ровно одно вычисление. Если вычисление является конечным, то его результатом объявляется состояние данных [t], отвечающее его последнему терму $t$. Две программы будем считать эквивалентными в модели $M$, если их вычисления в этой модели либо оба бесконечны, либо оба конечны и имеют одинаковый результат.

Под шкалой $\mathcal{F} = (S, s_0, R)$ будем понимать множество всех моделей вида $M = (S, s_0, R, \xi)$, где функция $\xi$ произвольна. Программы $\pi_1$, $\pi_2$ будем считать эквивалентными на шкале $\mathcal{F}$ ($\pi_1 \sim_{\mathcal{F}} \pi_2$), если они эквивалентны в любой модели, определяемой этой шкалой. Шкалу также можно расматривать как ориентированный помеченный граф с выделенным корнем $s_0$.

Шкалу $\mathcal{F} = (S, s_0, R)$ будем называть полугрупповой, если множество состояний шкалы $[h]$ с операцией $[h_1][h_2] = [h_1 h_2]$ образует полугруппу, и упорядоченной, если для любых базовых термов $t'$, $t''$ верно неравенство $[t't''] \neq [t']$. Для краткости будем отождествлять полугрупповую шкалу и описываемую ей полугруппу. Упорядоченную полугрупповую шкалу далее будем называть просто шкалой.

Введенные понятия позволяют поставить проблему эквивалентности программ следующим образом: для заданной шкалы $\mathcal{F}$ и заданной пары программ $\pi_1$, $\pi_2$ проверить выполнимость соотношения $\pi_1 \sim_{\mathcal{F}} \pi_2$.

Будем говорить, что программа $\pi = (F_\pi, D, T)$ представлена в нормальной форме, если $T = f \in F_\pi$, для всех $c \in C$, $f \in F_\pi$ терм $D(f, c)$ либо базовый, либо представим в виде $aft$, $a \in A$, $f \in F_\pi$, $t \in A^*$, в $F_\pi$ выделен специальный символ $f_{inf}$ такой, что $D(f_{inf}, c) = af_{inf}$ и все символы $F_\pi$, кроме $f_{inf}$, можно последовательностью замен $f$ на $D(f, c)$ привести к базовым термам. В [2] было показано, что любая программа может быть приведена к нормальной форме за полиномиальное время и с возрастанием сложности программы не более чем в полиномиальное число раз. Поэтому в дальнейшем считаем, что программы уже приведены к нормальной форме.

Четверку $K = (W, U, w^+, w^*)$, где $W$ — конечно порожденный моноид с операцией $*$ и нейтральным элементом $e$, $U$ — его подмоноид и $w^+, w^* \in W$, будем называть критериальной системой для шкалы $\mathcal{F}$, если

- существует гомоморфизм $\varphi : \mathcal{F} \times \mathcal{F} \to U$ такой, что $w^+ * \varphi(s_1, s_2) * w^* = e \Leftrightarrow s_1 = s_2$,
- уравнение $X * w = e$, где $w \in U * w^*$, имеет не более одного решения относительно $X$ и
- уравнение $w * X = e$, где $w \in w^+ * U$, имеет не более одного решения относительно $X$.

Пусть теперь заданы программы $\pi_1$, $\pi_2$, шкала $\mathcal{F}$ и критериальная система $K$ для этой шкалы. Опишем граф совместных вычислений $G_{\pi_1,\pi_2}$ программ $\pi_1$, $\pi_2$.

Вершинами графа являются четверки вида $(G_1, G_2, w, \hat{w})$, где $G_i \in F_{pi_i} \cup \{\lambda\}$, $w \in w^+ * U$, $\hat{w} \in U * w^*$. Произвольной паре термов вида $T_1 = t'_1 f_1 t''_1$, $T_2 = t'_2 f_2 t''_2$, где $t'_i, t''_i \in A^*$, $f_i \in F_{\pi_i} \cup \{\lambda\}$ (если $f_i = \lambda$, то $t''_i = \lambda$), соответствует вершина графа $V_{T_1,T_2} = (f_1, f_2, w^+ * \varphi([t'_1], [t'_2]), \varphi([t''_1], [t''_2]) * w^*)$.

Пусть $Tr_1 = Tr'_1 T_1 T'_1$, $Tr_2 = Tr'_2 T_2 T'_2$, где $T_i, T'_i \in Term$ — произвольные трассы программ $\pi_1$, $\pi_2$, реализуемые в некоторой общей модели (терм $T'_i$ отсутствует, если терм $T_i$ базовый), причем если $T_i = t'_i f_i t''_i$, то $c_i = \xi([t'_i])$. Дуги графа $G_{\pi_1,\pi_2}$ описываются четырьмя случаями. Если состояния $[t'_1]$, $[t'_2]$ совпадают или не достижимы

друг из друга, то из вершины $V_{T_1,T_2}$ в вершину $V_{T_1',T_2'}$ исходит дуга, несущая метку $(c_1, c_2)$. Если терм $T_1$ базовый или состояние $[t_1']$ достижимо из состояния $[t_2']$, то из вершины $V_{T_1,T_2}$ в вершину $V_{T_1,T_2'}$ исходит дуга, несущая метку $(\varepsilon, c_2)$. Если терм $T_2$ базовый или состояние $[t_2']$ достижимо из состояния $[t_1']$, то из вершины $V_{T_1,T_2}$ в вершину $V_{T_1',T_2}$ исходит дуга, несущая метку $(c_1, \varepsilon)$. Во всех остальных случаях из вершины $V_{T_1,T_2}$ не исходит никаких дуг.

Корнем графа объявляется вершина $(f_1, f_2, w^+, w^*)$, где $f_i$ — запрос программы $\pi_i$. Для простоты формулировок далее считаем, что граф $G_{\pi_1,\pi_2}$ содержит только вершины, достижимые из корня.

В графе $G_{\pi_1,\pi_2}$ особо выделяются опровергающие вершины и опровергающие циклы. Опровергающая вершина — вершина вида $(\lambda, \lambda, w, \hat{w})$, где $w * \hat{w} \neq e$. Опровергающий цикл — цикл, третьи компоненты всех вершин которого лежат в одном из множеств $U_\prec$, $U_\succ$, где $U_\prec = \{w^+ * \varphi(s_1, s_2) | s_1 \prec s_2\}$, $U_\succ = \{w^+ * \varphi(s_1, s_2) | s_2 \prec s_1\}$ (здесь запись $s' \prec s''$ означает, что состояние $s''$ достижимо из состояния $s'$ на шкале $\mathcal{F}$).

**Теорема.** *$\pi_1 \sim_{\mathcal{F}} \pi_2$ тогда и только тогда, когда в графе $G_{\pi_1,\pi_2}$ нет опровергающих вершин и опровергающих циклов.*

**Теорема.** *Если шкала $\mathcal{F}$ имеет критериальную систему $K$, то $\pi_1 \sim_{\mathcal{F}} \pi_2$ тогда и только тогда, когда граф $G_{\pi_1,\pi_2}$ не содержит опровергающих вершин и опровергающих циклов и его размер ограничен величиной $2^{O(n)}$, где $n = |\pi_1| + |\pi_2|$.*

**Теорема.** *Если шкала $\mathcal{F}$ имеет критериальную систему $K = (W, U, w^+, w^*)$, моноид $W$ является группой и проблемы достижимости состояний шкалы и равенства элементов моноида полиномиально разрешимы, то и проблема эквивалентности $\pi_1 \sim_{\mathcal{F}} \pi_2$ разрешима .*

Работа выполнена при поддержке ФЦП «Научные и научно-педагогические кадры инновационной России» 2009-2013 гг.

**Литература**

1. De Bakker J. W., Scott D. A. Theory of programs. Unpublished notes. // Vienna: IBM Seminar, – 1969.

2. Захаров В. А. Об эффективной разрешимости проблемы эквивалентности линейных унарных рекурсивных программ // Математические вопросы кибернетики, вып. 8. – М.: Наука, 1999. – с. 255-273.

# АЛГЕБРАИЧЕСКАЯ ХАРАКТЕРИЗАЦИЯ ЯЗЫКОВ, ДОПУСТИМЫХ В ОТМЕЧЕННЫХ ГРАФАХ

**Пряничникова Е.А. (Государственный университет информатики и искусственного интеллекта, Донецк, Украина )**
*Pryanichnikovae@gmail.com*

В теории конечных автоматов одним из важнейших результатов является теорема Клини, в которой утверждается, что класс языков, распознаваемых конечными автоматами, совпадает с классом рациональных языков, представимых регулярными выражениями алгебры Клини [1]. Основная цель данной работы - доказать аналогичную теорему для более широкого класса отмеченных графов и алгебр.

Графом с отмеченными дугами (конечным автоматом) назовем четверку $G = (Q, E, X, \mu)$, где $Q$ — конечное множество вершин; $E \subseteq Q \times Q$ — множество дуг; $X$ — конечное множество отметок дуг; $\mu : E \to X$ — функция отметок дуг.

Графом с отмеченными вершинами назовем четверку $G = (Q, E, X, \mu)$, где $Q$ — конечное множество вершин, $E \subseteq Q \times Q$ — множество дуг; $X$ — конечное множество отметок вершин; $\mu : Q \to X$ — функция отметок вершин.

Полностью отмеченным графом назовем четверку $G = (Q, E, X, \mu)$, где $Q$ — конечное множество вершин, $|Q| = n$; $E \subseteq Q \times Q$ — множество дуг; $X$ — конечное множество отметок; $\mu : Q \cup E \to X$ — функция отметок вершин и дуг.

Путем в графе будем называть конечную последовательность вершин $l = q_1 q_2 ... q_k$, где $(q_i, q_{i+1}) = e_i \in E$. Вершину $q_1$ будем называть начальной вершиной пути $l$, вершину $q_k$ - конечной вершиной пути.

Отметкой пути $l = q_1 q_2 ... q_k$ в графе с отмеченными дугами будем называть последовательность отметок входящих в этот путь дуг $\mu(e_1)\mu(e_2)...\mu(e_{k-1})$. Отметкой пути $l = q_1 q_2 ... q_k$ в графе с отмеченными вершинами будем называть последовательность отметок вершин $\mu(q_1)\mu(q_2)...\mu(q_k)$. Отметкой пути $l = q_1 q_2 ... q_k$ в полностью отмеченном графе будем называть чередующуюся последовательность отметок вершин и дуг $\mu(q_1)\mu(e_1)\mu(q_2)\mu(e_2)...\mu(q_k)$.

Пусть $I \subseteq Q$ — множество начальных вершин графа , $F \subseteq Q$ — множество финальных вершин. Отметки всех путей в отмеченном графе $G$, начальные вершины которых принадлежат множеству $I$, а

конечные — множеству $F$, назовем языком, допускаемым графом $G$, и обозначим $L(G)$.

Пусть $X$ — конечный алфавит; $X^*$ — множество всех слов конечной длины в алфавите $X$; $X^n$ — множество всех слов длины $n$ в алфавите $X$; $X^{\geq n}$ — множество всех слов конечной длины в алфавите $X$, длина которых больше или равна $n$.

Определим на множестве $X^*$ частичную бинарную операцию $\overset{n}{\circ}$ склеивания двух слов с параметром $n$ следующим образом: для всех $w_1, w_2 \in X^*$

$$w_1 \overset{n}{\circ} w_2 = \begin{cases} xyz, & \text{если } w_1 = xy,\ w_2 = yz,\ y \in X^n; \\ \text{не определено} & \text{в противном случае.} \end{cases}$$

Операция $\overset{n}{\circ}$ ассоциативна при любом $n$, то есть $(2^{X^*}, \overset{n}{\circ})$ и $(2^{X^{\geq n}}, \overset{n}{\circ})$ - полугруппы.

Нейтральный элемент по операции $\overset{n}{\circ}$ существует тогда и только тогда, когда она определена на множестве языков, в которых нет слов, длина которых меньше $n$. Если нейтральный элемент существует, то он равен $X^n$. Таким образом, полугруппа $(2^{X^*}, \overset{n}{\circ})$ является моноидом только при $n = 0$.

Введем на языках $L, R \subseteq X^*$ следующие операции:

1) $L \cup R = \{w : w \in L \text{ или } w \in R\}$;

2) $L \overset{n}{\circ} R = \{w_1 \overset{n}{\circ} w_2 : w_1 \in L \text{ и } w_2 \in R\}$;

3) $L^{\overset{n}{+}} = \bigcup\limits_{i=1}^{\infty} L^i$ , где $L^1 = L$; $L^{i+1} = L^i \overset{n}{\circ} L$ для всех $i \geq 1$.

Для характеризации языков, представимых в отмеченных графах, рассмотрим алгебры $(2^{X^*}, \overset{n}{\circ}, \cup, \overset{n}{+}, \emptyset)$ и $(2^{X^{\geq n}}, \overset{n}{\circ}, \cup, \overset{n}{*}, X^n, \emptyset)$.

Все алгебры $(2^{X^{\geq n}}, \overset{n}{\circ}, \cup, \overset{n}{*}, X^n, \emptyset)$ являются полукольцами.

Алгебра $(2^{X^*}, \overset{n}{\circ}, \cup, \overset{n}{+}, \emptyset)$ будет иметь единицу по операции $\overset{n}{\circ}$ только в случае, когда $n = 0$ и операция $\overset{n}{\circ}$ совпадает с конкатенацией, а рассматриваемая алгебра является алгеброй регулярных языков. Во всех остальных случаях эти алгебры не будут полукольцами.

Регулярные выражения в алгебре $(2^{X^*}, \overset{n}{\circ}, \cup, \overset{n}{+}, \emptyset)$ определим следующим образом:

1. $\varnothing$ является регулярным выражением и представляет язык $L(\varnothing) = \varnothing$;

2. $x$ является регулярным выражением и представляет язык $L(x) = \{x\}$ для всех $x \in \bigcup\limits_{0 \le i \le n+1} X^i$;

3. Если $R$ и $Q$ — регулярные выражения, представляющие языки $L(R)$ и $L(Q)$ соответственно, то выражения $(R \overset{n}{\circ} Q)$, $(R \cup Q)$, $(R^{\overset{n}{+}})$ также являются регулярными, причем $L(R \overset{n}{\circ} Q) = L(R) \overset{n}{\circ} L(Q)$, $L(R \cup Q) = L(R) \cup L(Q)$, $L(R^{\overset{n}{+}}) = (L(R))^{\overset{n}{+}}$.

**Теорема.** *Язык $L \subseteq X^*$ допустим в графе с отмеченными дугами, графе с отмеченными вершинами и полностью отмеченном графе тогда и только тогда, когда он описывается регулярным выражением любой алгебры из семейства $(2^{X^*}, \overset{n}{\circ}, \cup, \overset{n}{+}, \emptyset)$.*

Эта теорема в некотором смысле аналогична широко известной теореме Клини для конечных автоматов. В случае, когда $n = 0$ и рассматриваются только графы с отмеченными дугами, теорема 1 совпадает с теоремой Клини.

На основе доказательства теоремы разработаны методы анализа и синтеза языков, представимых в отмеченных графах.

Поскольку для описания одного и того же класса графов можно использовать различные алгебры, представляет интерес вопрос о связи таких алгебр между собой.

**Теорема.** *Для двух алгебр $(2^{X^*}, \overset{n_1}{\circ}, \cup, \overset{n_1}{+}, \emptyset)$ и $(2^{X^*}, \overset{n_2}{\circ}, \cup, \overset{n_2}{+}, \emptyset)$ в случае, когда $n_1 < n_2$ , существует такое отображение $\varphi : 2^{X^*} \to 2^{X^*}$ , которое является гомоморфизмом. Если $n_2 > n_1$, то гомоморфизма нет.*

Рассматриваемое в теореме отображение является инъекцией, поэтому в случае, когда $n_1 < n_2$ , алгебра $(2^{X^*}, \overset{n_1}{\circ}, \cup, \overset{n_1}{+}, \emptyset)$ изоморфно вложима в алгебру $(2^{X^*}, \overset{n_2}{\circ}, \cup, \overset{n_2}{+}, \emptyset)$, причем образ $\varphi$ является подалгеброй $(2^{X^*}, \overset{n_2}{\circ}, \cup, \overset{n_2}{+}, \emptyset)$ , а значит, все рассматриваемые алгебры входят в одно квазимногообразие, в которое входит алгебра регулярных языков.

**Теорема.** *Пусть $\Re(n)$ - множество всех регулярных выражений алгебры $(2^{X^*}, \overset{n}{\circ}, \cup, \overset{n}{+}, \emptyset)$ . Если $n_1 < n_2$ , то существует такое отображение $\psi : \Re(n_1) \to \Re(n_2)$ , которое сохраняет язык регулярного выражения, то есть, если $r$ - это регулярное выражение*

*алгебры* $(2^{X^*}, \overset{n_1}{\circ}, \cup, \overset{n_1}{+}, \emptyset)$, $L(r)$ - *язык, представляемый этим регулярным выражением, то* $\psi(r)$ - *это регулярное выражение алгебры* $(2^{X^*}, \overset{n_2}{\circ}, \cup, \overset{n_2}{+}, \emptyset)$ *и* $L(\psi(r)) = L(r)$

В данной работе рассматриваются языки, допустимые в отмеченных графах: графах с отмеченными дугами, графах с отмеченными вершинами и графах, в которых отмечены и дуги, и вершины. Найдена алгебраическая характеризация таких языков, разработаны методы их анализа и синтеза. Исследованы основные свойства семейства алгебр языков, допустимых в отмеченных графах.

**Литература**

1. Anderson J. Automata Theory with Modern Applications. — Cambridge: Cambridge University Press, 2006.

2. Капитонова Ю. В., Летичевский А. А. Математическая теория проектирования вычислительных систем. — М.: Наука, 1988.

# БЫСТРЫЙ АЛГОРИТМ ПОСТРОЕНИЯ ДЛЯ $k$-ЗНАЧНЫХ ФУНКЦИЙ ПОЛИНОМОВ ПО СОСТАВНОМУ МОДУЛЮ $k$

**Селезнева С.Н. (МГУ имени М.В. Ломоносова)**
*e-mail: selezn@cs.msu.su*

## Введение

Рассматривается задача проверки полиномиальности $k$-значных функций (функций над кольцами вычетов по модулю $k$). Известно, что каждая $k$-значная функция может быть задана полиномом по модулю $k$ в том и только в том случае, если если $k$ – простое число[1]. Селезневой С.Н. в [2] был предложен практически применимый алгоритм, который по вектору значений $k$-значной функции $f(x_1, \ldots, x_n)$, где $k = p^m$, $p$ – простое число, $m \geq 2$, определяет, задается ли $f$ полиномом по модулю $k$, и в случае положительного ответа находит ее канонический полином, причем алгоритм имеет битовую сложность $O(N)$, где $N = k^n$ – длина вектора значений функции. Этот алгоритм можно обобщить на случай произвольного составного числа $k$, при этом его сложность остается такой же.

В настоящей заметке подробно рассматривается случай произвольного составного $k$. Приведено описание алгоритма, который по вектору значений $k$-значной функции $f(x_1, \ldots, x_n)$ определяет, задается ли эта функция полиномом по модулю $k$, в случае положительного ответа строит один из ее полиномов, причем алгоритм имеет битовую сложность $O(N)$, где $N = k^n$ – длина вектора значений функции $f$.

## Основные понятия

Пусть $k \geq 2$, $E_k = \{0, 1, \ldots, k-1\}$. Функция $f(x_1, \ldots, x_n)$ называется *$k$-значной*, если $f : E_k^n \to E_k$, где $n = 1, 2, \ldots$.

Множество всех $k$-значных функций обозначим как $P_k$, множество всех $k$-значных функций, зависящих от переменных $x_1, \ldots, x_n$, обозначим как $P_k^n$.

Пусть $\mathbb{Z}$ – множество целых чисел; $\mathbb{Z}_k = \mathbb{Z}/(k) = \{0, 1, \ldots, k-1\}$ – кольцо вычетов по модулю $k$, где $k \geq 1$.

Функция $f(x_1, \ldots, x_n) \in P_k^n$ *задается полиномом по модулю $k$*, если найдется такой полином $p(x_1, \ldots, x_n) \in \mathbb{Z}_k[x_1, \ldots, x_n]$, что

$$p(x_1, \ldots, x_n) = f(x_1, \ldots, x_n).$$

Множество $k$-значных функций, задающихся полиномами по модулю $k$, обозначим как $Pol_k$, и будем называть их полиномиальными.

Известно [1], что $Pol_k = P_k$ тогда и только тогда, когда $k$ – простое число.

Быстрые алгоритмы построения по вектору значений $k$-значной функции $f(x_1, \ldots, x_n)$ ее полинома по модулю $k$ при простых $k$ предложены Гавриловым Г.П., Сапоженко А.А. (1977 г. [3]) для $k = 2$, Тарannиковым Ю.В. (2004 г.) для произвольного простого числа $k$. Сложность этих алгоритмов (в алгоритмической модели СФЭ) равна $O(N \log N)$ битовых операций, где $N = k^n$ – длина вектора значений функции.

Мещаниновым Д.Г. (1995 г. [4]) описан алгоритм, проверяющий по вектору значений $k$-значной функции $f(x_1, \ldots, x_n)$ для $k = p^m$, где $p$ – простое число, $m \geq 2$, является ли функция $f$ полиномиальной и в случае положительного ответа строящий какой-то ее полином. Сложность этого алгоритма равна $O(N \log^m N)$ операций, где $N = k^n$ – длина вектора значений функции.

Селезневой С.Н. (2011 г. [2]) предложен алгоритм, проверяющий по вектору значений $k$-значной функции $f(x_1, \ldots, x_n)$ для $k = p^m$, где $p$ – простое число, $m \geq 2$, является ли функция $f$ полиномиальной и в случае положительного ответа строящий ее канонический полином. Сложность этого алгоритма (в алгоритмической модели СФЭ) равна $O(N)$ битовых операций, где $N = k^n$ – длина вектора значений функции. Этот алгоритм обобщается на случай произвольного составного $k$, и сложность его остается такой же.

В настоящей заметке опишем этот алгоритм для случая произвольного составного числа $k$.

## Алгоритм распознавания полиномиальности и построения полиномов

В качестве алгоритмической модели рассмотрим схемы из функциональных элементов (СФЭ) в некотором полном в $P_k$ базисе. Под сложностью алгоритма будем понимать число функциональных элементов в соответствующей СФЭ.

**Теорема**. *Пусть $k$ – составное число. Можно построить детерминированный алгоритм (в алгоритмической модели СФЭ), который для произвольной функции $f(x_1, \ldots, x_n) \in P_k^n$ по вектору ее значений определяет, верно ли, что $f \in Pol_k^n$, и в случае по-*

*ложительного ответа строит какой-то ее полином со сложностью $O(N)$ битовых операций (с числом функциональных элементов $O(N)$), где $N = k^n$ – длина вектора значений функции.*

**Доказательство**. Опишем алгоритм, который для произвольной функции $f \in P_k^n$ по вектору ее значений определяет, является ли она полиномиальной, и в случае положительного ответа строит какой-то ее полином.

Пусть $k$ – составное число, $k = p_1^{m_1} \cdot \dots \cdot p_r^{m_r}$, где $p_i$ – попарно различные простые числа, $m_i \geq 1$.

Если $r = 1$, т.е. $k = p^m$, где $p$ – простое число, $m \geq 2$, – воспользуемся алгоритмом из [2].

Пусть $r \geq 2$. Тогда кольцо $\mathbb{Z}_k$ есть прямая сумма идеалов, изоморфных кольцам $\mathbb{Z}_{p_i^{m_i}}$.

**Шаг 1**. Пусть $d_i = p_i^{m_i}$, $i = 1, \dots, r$.

Пусть $\alpha = (a_1, \dots, a_n) \in E_k^n$ и $\beta = (b_1, \dots, b_n) \in E_k^n$. Будем говорить, что наборы $\alpha$ и $\beta$ *сравнимы по модулю* $d_i$ и обозначать $\alpha = \beta (\mathrm{mod}\ d_i)$, если

$$a_1 = b_1 (\mathrm{mod}\ d_i), \dots, a_n = b_n (\mathrm{mod}\ d_i).$$

По свойствам полиномов над кольцами вычетов по модулю $k$ если функция $f(x_1, \dots, x_n) \in Pol_k$, то для любых наборов $\alpha, \beta \in E_k^n$ из того, что $\alpha = \beta (\mathrm{mod}\ d_i)$, следует $f(\alpha) = f(\beta) (\mathrm{mod}\ d_i)$.

Выполним проверку этого необходимого условия полиномиальности для функции $f(x_1, \dots, x_n)$.

Для этого для каждого $i$, $i = 1, \dots, r$, сравниваем значения функции $f$ на наборах, сравнимых по модулю $p_i^{m_i}$. Т.к. сравнимость наборов по модулю $p_i^{m_i}$ задает отношение эквивалентности на множестве $E_k^n$, то проверку для каждого $i$ можно выполнить со сложностью $O(N)$. Значит, сложность шага 1 будет также $O(N)$ битовых операций.

Если хотя бы однажды условие не выполняется, то $f \notin Pol_k$. Иначе, переходим к шагу 2.

**Шаг 2**. По китайской теореме об остатках сопоставим каждому элементу $a \in E_k$ однозначный набор $(a_1, \dots, a_r) \in E_{p_1^{m_1}} \times \dots \times E_{p_r^{m_r}}$, являющийся решением системы сравнений:

$$a = a_i (\mathrm{mod}\ p_1^{m_1}), \dots, a = a_r (\mathrm{mod}\ p_r^{m_r}).$$

В силу выполненного условия шага 1 функции $f(x_1,\dots,x_n) \in P_k^n$ сопоставляется набор функций $(f_1,\dots,f_r) \in P^n_{p_1^{m_1}} \times \cdots \cdot P^n_{p_1^{m_1}}$.

Теперь для каждого $i$, $i = 1,\dots,r$, надо проверить задается ли функция $f_i(x_1^i,\dots,x_n^i) \in P^n_{p_i^{m_i}}$ полиномом по модулю $p_i^{m_i}$ и в случае положительного ответа найти какой-то ее полином.

Воспользуемся известными алгоритмами.

Если $m_i = 1$, то применим алгоритмы построения полиномов при простых $p_i$ (Гаврилов Г.П., Сапоженко А.А.[3], Таранников Ю.В.). Получим вектор коэффициентов полинома функции $f_i$ со сложностью $O(p_i^{m_i n} n)$.

Если $m_i \geq 2$, то применим алгоритмы распознавания полиномиальности и построения полиномов при составных $p_i^{m_i}$ (Селезнева С.Н. [2]). Если $f_i \notin Pol_{p_i^{m_i}}$, то $f \notin Pol_k$. Иначе, получим вектор коэффициентов канонического полинома функции $f_i$ со сложностью $O(p_i^{m_i n})$.

Сложность шага 2 равна $O(N)$.

**Шаг 3**. Если все функции $f_i$ – полиномиальны (каждая по своему модулю), то опять-таки по китайской теореме об остатках по набору коэффициентов $c_1 \in E_{p_1^{m_1}},\dots,c_r \in E_{p_r^{m_r}}$ при мономе $X$ в полиномах функций $f_1,\dots,f_r$ соответственно найдем коэффициент $c \in E_k$ при мономе $X$ в полиноме функции $f$, являющийся решением системы сравнений:

$$c = c_1 (\mathrm{mod}\ p_1^{m_1}),\dots,c = c_r (\mathrm{mod}\ p_r^{m_r}).$$

Выполнить шаг 3 можно со сложностью $O(N)$.

Теорема доказана.

Работа поддержана РФФИ, гранты 09-01-00701а, 10-01-00768а.

**Литература**

1. Яблонский С.В. Функциональные построения в $k$-значной логике. Труды МИАН (1958) 51, с. 5-142.

2. Селезнева С.Н. Быстрый алгоритм построения для $k$-значных функций полиномов по модулю $k$ при составных $k$. Дискретная математика (2011) 23, вып. 3, с. 3-22.

3. Гаврилов Г.П., Сапоженко А.А. Сборник задач по дискретной математике. М.: Наука, 1977.

4. Мещанинов Д.Г. Метод построения полиномов для функций $k$-значной логики. Дискретная математика (1985) 7, вып. 3, с. 48-60.

# ПОРЯДОК ФУНКЦИИ ШЕННОНА ДЛЯ НАКОПЛЕННОГО ВЕТВЛЕНИЯ СХЕМ ИЗ ФУНКЦИОНАЛЬНЫХ ЭЛЕМЕНТОВ

Стариков А. О.
(Московский государственный университет имени М. В. Ломоносова, студент)
*E-mail: alexey.starikov@mail.ru*

**Введение**

Одной из основных задач синтеза схем из функциональных элементов (СФЭ) является синтез схем, минимальных относительно некоторой заданной характеристики. Для оценки качества конкретных алгоритмов синтеза бывает полезным сопоставление ассимптотики сложности данного алгоритма с функцией Шеннона для минимальной схемы. Ассимптотика функции Шеннона для сложности, представляемой в виде количества элементов в схеме в стандартном базисе, была найдена О. Б. Лупановым в работе [4]. Ассимптотика функции Шеннона для сложности, заданной глубиной схемы в произвольном бесконечном базисе, была найдена О. М. Касим-Заде в работе [5].

Исходя из некоторых свойств практической реализации схем из функциональных элементов [7], имеет смысл рассмотреть характеристику, связанную с топологией соединения элементов схемы. В данной работе исследуется функция Шеннона для сложности, представленной в виде накопленного ветвления, то есть суммарного ветвления провода по пути от входа схемы к её выходу. В качестве базиса рассматривается стандартная система $\{\vee, \&, \neg\}$, дополненная тождественной функцией $\{x\}$.

Как оказалось, методы синтеза Шеннона [1] и Лупанова [4] дают схемы с большим накопленным ветвлением (порядка не меньше $O(2^n)$). При этом построенный автором метод, основанный на «балансировке» совершенной ДНФ с помощью вставок древовидных схем из тождественных функций, дал достаточно хорошую оценку. В работе этот метод используется для получения верхней оценки значения функции Шеннона для накопленного ветвления. Автором получена также нижняя оценка, дающая в совокупности с верхней порядковую оценку $O(n)$, при этом верхняя оценка отличается от нижней менее чем в два раза. Для доказательства нижней оценки применяются мощностные соображения, приведенные в доказательстве ниж-

ней оценки функции Шеннона для количества элементов СФЭ [1].

**Основные понятия и результаты**

Будем исходить из определения схемы из функциональных элементов как нагруженного ориентированного графа.

Вершины ориентрованного графа, в которые не входит ни одного ребра, называются истоками. Орграф называется ациклическим, если в нем нет ориентированных циклов. В ациклическом орграфе глубиной вершины $\nu$ называется максимальное число ребер в ориентированном пути из какого-либо истока в вершину $\nu$. Орграф называется упорядоченным, если для каждой вершины $\nu_i$, в которую входит $k_i$ ребер, задан порядок $e_1, e_2, \ldots, e_{k_i}$ этих ребер.

Систему $Б = \{g_1, g_2, \ldots, g_m\}$, где все $g_i$ — функции алгебры логики, будем называть базисом функциональных элементов. В дальнейшем, как правило, будем подразумевать под базисом функциональных элементов систему $Б_0 = \{\vee, \&, \neg, x\}$. Так как все эти функции симметричны относительно своих переменных, то ребра, входящие в каждую вершину, можно не упорядочивать.

Схемой из функциональных элементов (СФЭ) над базисом $Б$ называется ациклический упорядоченный орграф, в котором:

1) каждому истоку приписана некоторая переменная, причем разным истокам приписаны разные переменные (истоки при этом называются входами схемы, а приписанные им переменные — входными переменными);

2) каждой вершине, в которую входят $k \geq 1$ ребер, приписана функция из базиса $Б$, зависящая от $k$ переменных (вершина с приписанной функцией при этом называется функциональным элементом);

3) некоторые вершины выделены как выходы (истоки также могут являтся выходами).

Индукцией по глубине $q$ вершины $\nu$ определяется функция $f_\nu$, реализуемая в данной вершине. Если $q = 0$, то есть $\nu$ — исток, и $\nu$ приписана переменная $x_i$, то $f_\nu \equiv x_i$. Пусть реализуемые функции уже определены для всех вершин глубины меньшей, чем $q_0$. Рассмотрим вершину $\nu$ глубины $q_0$, в которую входят ребра $e_1, e_2, \ldots, e_k$ из вершин $\nu_1, \nu_2, \ldots, \nu_k$, и в этих вершинах реализуются функции $f_1, f_2, \ldots, f_k$. Пусть вершине $\nu$ приписана функция $g(x_1, \ldots, x_k)$. Тогда в $\nu$ реализуется функция $f_\nu = g(f_1, f_2, \ldots, f_k)$.

Будем говорить, что схема реализует систему функций, реализуемых в ее выходах. Схема реализует данную функцию, если она реализует ее хотя бы на одном из выходов.

Сложностью схемы из функциональных элементов называется число функицональных элементов в схеме. Будем обозначать сложность через $L$.

Назовем проводом, исходящим из функционального элемента $\nu$, множество вершин $\{\nu_1, \ldots, \nu_F\}$, в которые из элемента $\nu$ идут ребра. Ветвлением провода назовем число вершин $F$ в соответствующем ему множестве. Обозначим ветвление провода, исходящего из вершины $\nu$, через $F(\nu)$. Будем считать, что если из $\nu$ не выходит ни одного ребра, то $F(\nu) = 1$.

При этом будем считать, что ветвление входов запрещено, то есть для любого истока $\nu_0$ справедливо $F(\nu_0) = 1$.

Пусть $P = (\nu_0, \nu_1, \nu_2, \ldots, \nu_n)$ — путь из истока $\nu_0$ (входной вершины) в выходную вершину $v_n$. Назовем накопленным ветвлением по пути $P$ величину

$$F(P) = \sum_{i=1}^{n} F(\nu_i).$$

Назовем накопленным ветвлением СФЭ $S$ величину

$$F(S) = \max_{P\text{ — путь из входа в выход}} F(P).$$

Введем функцию Шеннона для накопленного ветвления СФЭ по формуле

$$F(n) = \max_{f \in P_2(n)} \quad \min_{f \text{ реализуется СФЭ } S} \quad F(S).$$

Автором получены следующие оценки функции Шеннона для накопленного ветвления:

**Теорема 1.** *Для схем из функциональных элементов над базисом* $Б_0 = \{\vee, \&, \neg, x\}$

$$F(n) \lesssim ](3\log_3 2 + 1)n[ + ]\log_2 n[ + 1.$$

**Теорема 2.** *Для схем из функциональных элементов над базисом* $Б_0 = \{\vee, \&, \neg, x\}$

$$F(n) \gtrsim 3(\log_3 2)n - 6(\log_3 2)\log_2 n - 3.$$

Из теорем 1 и 2 непосредственно следует

**Следствие 1.** *Порядок функции Шеннона $F(n)$ над базисом $Б_0$ равен $O(n)$.*

Кроме того, нижняя оценка функции Шеннона для накопленного ветвления СФЭ существенна:

**Следствие 2.** *Для почти всех функций из $P_2$ порядок наименьшего накопленного ветвления среди схем из функциональных элементов, реализующих данную функцию, совпадает с порядком функции Шеннона.*

Автор выражает благодарность своему научному руководителю Игорю Викторовичу Кучеренко за постановку задачи и внимание к работе и академику Валерию Борисовичу Кудрявцеву за ценные советы и замечания.

**Литература**

1. Яблонский С. В. Введение в дискретную математику. – М.: Высшая школа, 2002.

2. Яблонский С. В., Гаврилов Г. П., Кудрявцев В. Б. Функции алгебры логики и классы Поста. – М.: Наука, 1966.

3. Кудрявцев В. Б., Блохина Г. Н., Кнап Ж., Кудрявцев В. В. Алгебра логики. – Москва-Любляна: Издательство механико-математического факультета МГУ, 2006.

4. Лупанов О. Б. О синтезе некоторых классов управляющих систем // Проблемы кибернетики. – М.: Физматгиз, 1963, вып. 10, с. 63-97.

5. Касим-Заде О. М. О глубине булевых функций над произвольным бесконечным базисом // Дискретный анализ и исследование операций, сер. 1. – 2007, том 14, №1, с. 45–69.

6. Naveed A. Sherwani. Algorithms for VLSI Physical Design Automation. Third Edition. – Springer, 1998.

7. Lawrence T. Pillage, Ronald A. Rohrer. Asymptotic waveform evaluation for timing analysis // IEEE Trans. on CAD of Integrated Circuits and Systems, 1990, 9, №4, с. 352–366.

# О МИНИМИЗАЦИИ СЛОЖНОСТИ ПРЕДСТАВЛЕНИЯ БУЛЕВЫХ ФУНКЦИЙ ИЗ НЕКОТОРЫХ КЛАССОВ

**Чебурахин И.Ф. (г. Москва)**
*cybernetics@mati.ru*

**Введение.** Рассматривается задача реализации булевых функций (БФ) в классе формул и — схем из функциональных элементов (ФЭ) в базисе Жегалкина, а также получения при этом по возможности минимальных значений показателей сложности. От сложности-качества этих схем зависят основные характеристики вычислительных и управляющих систем. Проводимые исследования в этой области свидетельствуют, что получение минимального решения неизбежно предполагает использование алгоритмов переборного характера. Следствием этого является большая трудоемкость поиска такого решения уже для функций небольшой размерности. Это приводит к разработке новых подходов постановки задачи и её решения, заметно отличающихся по трудоёмкости от переборных [1–3].

Символ $\cdot$ (точка) используется для обозначения логического или арифметического умножения [4-6].

**1. Булевы функции, базисы, формулы и схемы из ФЭ, показатели сложности.** Пусть $f$ ($f^{(n)}$ или $f(X)$) — булева функция, зависящая от $n$ переменных из множества $X = \{x_1, \ldots, x_n\}$. Под базисом $G$ понимаем конечную функционально полную систему БФ (или соответствующих ФЭ), в частности, — $G = \{\&, \oplus, 1\}$ для всех булевых функций. Считаем, что функция $f^{(n)}$ задается формулой $F^{(n)}$ в базисе $G$. В качестве меры сложности представления функции $f$ формулой $F$ или схемой $S$ из ФЭ определяем соответствующие показатели (дискретные функционалы): $L_a(f, G)$ — суммарное число вхождений символов переменных в формулу $F$; $L_F(f, G)$ — число базисных подформул в $F$; $Dep_F(f, G)$ — глубина $F$; $L_S(f, G)$ — число ФЭ в схеме $S$; $Dep_S(f, G)$ — глубина схемы $S$.

По практическим соображениям показатели сложности минимизируем. При представлении БФ в классе формул (включая скобочные) для минимизации показателей сложности используются эквивалентные преобразования, — в классе схем для минимизации числа ФЭ дополнительно применяется ветвление их выходов [4–9].

**2. Функциональные уравнения (ФУ).** Напомним определение ФУ типа 1 [4–9]: $f^{(n)} = h(f^{(n-1)}, x_n)$, где $n \geqslant 2$, $f^{(2)}$ — начальная

функция, $h^{(2)}$ — функция рекурсии, входящая в базис $G$ или представляемая через базисные функции. Обобщим этот тип ФУ. Для полинома Жегалкина $F^{(n)}$ определяем вектор $p$ повторяемости переменных множества $X = \{x_1, \ldots, x_i, \ldots, x_n\}$ в формуле $F^{(n)}$, то есть $p = (p_1, \ldots, p_i, \ldots, p_n)$, где переменная $x_i$ повторяется в формуле $F^{(n)}$ число $p_i$ раз. Заодно получаем $L_a(F^{(n)}, G) = \sum_{i=1}^{n} p_i$.

Пусть $p_i = \max\{p_1, \ldots, p_i, \ldots, p_n\}$, тогда ФУ типа 1 имеет вид

$$F^{(n)} = \left(\left(x_i \cdot F^{(n-1),1}\right) \oplus F^{(n-1),2}\right), \tag{1}$$

где верхние индексы 1 и 2 — номера соответствующих остаточных подфункций, зависящих от числа $(n-1)$ переменных. На основе этого ФУ строится алгоритм градиентного типа, позволяющий получить требуемую формулу $F^{(n)}$. С помощью (1) получаем верхнюю оценку сложности $L_F$. На каждом шаге алгоритма для соответствующей переменной $x_i$ $(1 \leqslant i \leqslant n-2)$ применяется не больше двух базисных операций и не более двух остаточных подфункций. Для оставшихся подформул, зависящих от переменных $x_{n-1}$ и $x_n$, при их представлении может потребоваться не более четырёх базисных функций. Итого, получаем $L_F(F^{(n)}, G) \leqslant 2^n$.

**3. Элементарные симметрические полиномы (ЭСП) Жегалкина [7, 8].** Рассмотрим ЭСП Жегалкина $F_i(\mathrm{n})$, где $n$ — число переменных, $i$ — степень полинома, т. е. $2 \leqslant n \leqslant N$, $1 \leqslant i \leqslant n$.

$$\begin{aligned}
&F_1^{(n)}(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n,\\
&F_2^{(n)}(x_1, \ldots, x_n) = x_1 \cdot x_2 \oplus x_1 \cdot x_3 \oplus \ldots \oplus x_{n-1} \cdot x_n,\\
&\ldots\\
&F_n^{(n)}(x_1, \ldots, x_n) = x_1 \cdot \ldots \cdot x_n.
\end{aligned} \tag{2}$$

При $i = 1$ или $n$ получаем классы функций «$\oplus$» и «&» (т.е. $F_1^{(n)}$ или $F_n^{(n)}$), для которых получены следующие совпадающие оценки

$L_a(F_1^{(n)}, G)_{\min} = L_a(F_n^{(n)}, G)_{\min} = n$ , $L_F(F_1^{(n)}, G)_{\min} =$
$= L_S(F_1^{(n)}, G)_{\min} = L_F(F_n^{(n)}, G)_{\min} = L_S(F_n^{(n)}, G)_{\min} = n - 1,$
$Dep_F(F_1^{(n)}, G)_{\min} = Dep_S(F_1^{(n)}, G)_{\min} = Dep_F(F_n^{(n)}, G)_{\min} =$
$= Dep_S(F_n^{(n)}, G)_{\min} = \lceil \log_2 n \rceil.$

При помощи ФУ типов 1 или 2 получены и другие оценки показателей сложности, из которых ниже потребуются следующие [4–8]:

$$L_S(F_2^{(n)},G)_{\min}=3n-5,\qquad L_S(F_3^{(n)},G)_{\min}=5n-13,$$
$$L_S(F_4^{(n)},G)=7n-25,\qquad L_S(F_5^{(n)},G)=9n-41,\qquad (3)$$
$$L_S(F_6^{(n)},G)=11n-61.$$

Приводя ФУ (1) к виду $F_i^{(n+1)}=F_i^{(n)}\oplus(x_{n+1}\cdot F_{i-1}^{(n)})$, где $n\geqslant 2$, $2\leqslant i\leqslant n-1$, удобно записывать ЭСП $F_i^{(n)}$ при помощи таблицы [9].

Для ЭСП Жегалкина $F_i^{(n)}$, где $n$ — число переменных и $i$ — степень полинома, поставим задачу вывода оценки $L_S(F_i^{(n)},G_3)=L_S(i,n)=U(i,n)$ для $n\geqslant 2$, $1\leqslant i\leqslant n$. Для имеющихся оценок сложности $U(i,n)$ (3) выполним преобразования, разбивая каждую из них на три алгебраические слагаемые (кроме первой и второй оценок). Итак, из каждой $U(i,n)$, $3\leqslant i\leqslant n-1$, выделяем первое слагаемое $(n-1)$, затем из оставшегося выражения выделяем произведение $(i-1)\cdot(2n-4)$. Тогда оставшаяся часть — сеточная функция $u_i$ ($u_i$: 4, 12, 24, 40, ..., для значений аргумента $i$, $3\leqslant i\leqslant n-1$), получается вычитанием из исходного выражения первых двух слагаемых:

$U(1,n)=L_S(F_1^{(n)},G)=n-1,$
$U(2,n)=L_S(F_2^{(n)},G)=3n-5=(n-1)+(2n-4),$
$U(3,n)=L_S(F_3^{(n)},G)=5n-13=(n-1)+2(2n-4)-4,$
$U(4,n)=L_S(F_4^{(n)},G)=7n-25=(n-1)+3(2n-4)-12,$
$U(5,n)=L_S(F_5^{(n)},G)=9n-41=(n-1)+4(2n-4)-24,$
$U(6,n)=L_S(F_6^{(n)},G)=11n-61=(n-1)+5(2n-4)-40,\ldots$

Для функции $u_i$ составляем разности первого, второго и далее порядков, пока не получим нулевую строку (если существует). Из того, что разности третьего порядка равны нулю, следует второй порядок для многочлена $u_i$ с неопределенными коэффициентами, т. е.

$$u_i=a_0\cdot i^2+a_1\cdot i+a_2. \qquad (4)$$

Решаем систему уравнений, получаемую из (4) для $i=3,\ 4,5$:

$$a_0\cdot 5^2+a_1\cdot 5+a_2=24,$$
$$a_0\cdot 4^2+a_1\cdot 4+a_2=12,$$
$$a_0\cdot 3^2+a_1\cdot 3+a_2=4.$$

Находим $a_0 = 2$; $a_1 = -6$; $a_2 = 4$. Таким образом, функция $u_i = 2 \cdot i^2 - 6 \cdot i + 4$. С третьей составляющей искомая оценка сложности

$$U(i,n) = L_S(F_i^{(n)}, G) = (n-1)+(i-1)\cdot(2\cdot n-4)-(2\cdot i^2-6\cdot i+4). \quad (5)$$

Итак, аналитически получена верхняя оценка показателя $L_S(F_i^{(n)}, G)$.

Для оценок (5) высказывается гипотеза, справедливая для показателей $L_S(F_1^{(n)}, G)$, и $L_S(F_n^{(n)}, G)$: значения показателя $L_S$ сложности, получаемые при помощи функционала (5) минимальные.

**Литература**


1. Журавлев Ю. И.Теоретико-множественные методы в алгебре логики // Проблемы кибернетики. 1962. №8.

2. Лупанов О. Б. О сложности реализации функций алгебры логики формулами // Проблемы кибернетики. Вып. 3. М.: 1960.

3. Яблонский С. В. Об алгоритмических трудностях синтеза минимальных контактных схем // Проблемы кибернетики, №2. М.: 1959.

4. Чебурахин И. Ф. Функциональные уравнения и сложность произвольной булевой функции в разных базисах // 7-я Межд. научн. конф. «Дискретные модели в теории управляющих систем». М.: 2006.

5. Чебурахин И. Ф. Преобразования функциональных уравнений и показатели сложности булевых функций // Материалы IX Межд.семинара «Дискретная математика и её приложения». М.: 2007.

6. Чебурахин И. Ф. Математические модели для интеллектуализации синтеза дискретных логических управляющих устройств на основе цифровых интегральных схем // Изв. РАН. ТиСУ. №1. 2008.

7. Чебурахин И. Ф. Показатели сложности симметрических полиномов Жегалкина. // Тез.докл. XV Межд.конф.«Проблемы теоретической кибернетики». Казань. 2008.

8. Чебурахин И. Ф. Сложность симметрических полиномов Жегалкина // XVII Межд. школа-семинар «Синтез и сложность управляющих систем». М.: 2008.

9. Чебурахин И. Ф., Цурков В. И. Синтез дискретных логических устройств обработки информации на основе теории агентов. // Мехатроника, автоматизация, управление. №3. 2011. - С. 27-34.

# СЛОИСТОСТЬ БУЛЕВЫХ ФУНКЦИЙ И ФУНКЦИЙ k-ЗНАЧНОЙ ЛОГИКИ

**Членова Т.С. (МГУ им М.В. Ломоносова)**
*tatiana-tch@mail.ru*

В статье изучается понятие слоистости булевых функций и функций $k$-значной логики над конечными полными системами. В определенном смысле слоистость функции можно понимать как ее глубину над бесконечной полной системой специального вида. Глубина функций над бесконечными полными системами общего вида исследовалась О.М. Касим-Заде и А.В. Кочергиным в работах [1], [2].

Введем понятие слоистости функций $k$-значной логики над полными системами в $P_k$, а также слоистости полных систем в $P_k, k \geq 2$.

Пусть $G = \{g_1, \ldots, g_n\}$ - полная система в $P_k, k \geq 2$. Назовем блоком над $\{g_i\}$ схему с одним выходом, построенную из элементов $g_i$ с помощью операций суперпозиции. Обозначим $B_i = \{B | B-$блок над$\{g_i\}\}$, $G_i = \{g | g \in P_k, g$ реализуется схемой из $B_i\}$ и $\widetilde{G} = \cup_{i=1}^{n} G_i$.

*Определение* 1. Слоистостью схемы $\phi$ с одним выходом над системой $\widetilde{G}$ в $P_k, k \geq 2$ назовем число $S_G(\phi)$, равное глубине этой схемы.

*Определение* 2. Слоистостью функции $f \in P_k$ над полной системой $G$ в $P_k, k \geq 2$ называется число $S_G(f) = \min_{\phi \in \Phi} S_G(\phi)$ , где $\Phi$ - множество всех схем над системой $\widetilde{G}$, реализующих функцию $f$.

*Определение* 3. Слоистостью полной системы $G$ в $P_k, k \geq 2$ будем называть число $S(G)$, равное максимуму слоистостей всех функций $f \in P_k$ над $G$, если множество чисел $\{S_G(f) | f \in P_k\}$ ограничено. Если же это множество чисел является неограниченным, то будем считать слоистость системы равной бесконечности.

Наша цель - исследование слоистостей полных системы в $P_k, k \geq 2$. Начнем со случая $P_2$.

Доказана следующая теорема.

**Теорема 1.** *Пусть $G$ - полная система в $P_2$. Тогда $S(G) \leq 4$.*

Используя понятия из [3], введем обозначения следующих множеств:

$Sh$ - множество всех Шефферовских функций;

$T_0$ - множество всех функций, сохраняющих 0;

$T_1$ - множество всех функций, сохраняющих 1;

$M$ - множество всех монотонных функций;

$L$ - множество всех линейных функций;

$S$ - множество всех самодвойственных функций;
$A_0 = \{f | f \in P_2, f(x, \ldots, x) = 0\}$;
$A_1 = \{f | f \in P_2, f(x, \ldots, x) = 1\}$;
$A_x = \{f | f \in P_2, f(x, \ldots, x) = x\}$;
$A_{\overline{x}} = \{f | f \in P_2, f(x, \ldots, x) = \overline{x}\}$;
$P_2^{(1)}$ - множество всех булевских функций, существенно зависящих не более, чем от одной переменной.

В следующих случаях получены более точные оценки слоистости полных систем в $P_2$:

I. Для полной системы $G$ такой, что $G \cap Sh \neq \emptyset$, выполнено равенство $S(G) = 1$;

II. Для полной системы $G$, обладающей одним из следующих свойств:

1. $G \cap Sh = \emptyset$, $G \cap A_0 \neq \emptyset$, $G \cap A_1 \neq \emptyset$, $G \setminus (M \cup L) \neq \emptyset$;
2. $G \cap Sh = \emptyset$, $G \cap A_0 \neq \emptyset$, $G \cap A_1 = \emptyset$, $(G \cap (A_{\overline{x}} \setminus L)) \neq \emptyset$;
3. $G \cap Sh = \emptyset$, $G \cap A_0 = \emptyset$, $G \cap A_1 \neq \emptyset$, $(G \cap (A_{\overline{x}} \setminus L)) \neq \emptyset$,

выполнено равенство $S(G) = 2$;

III. Для полной системы $G$, обладающей одним из следующих свойств:

1. $G \cap A_0 \neq \emptyset$, $G \cap A_1 \neq \emptyset$, $(G \setminus M) \setminus P_2^{(1)} \neq \emptyset$;
2. $G \cap A_0 \neq \emptyset$, $G \cap A_1 = \emptyset$, $(G \cap (A_0 \setminus L)) \neq \emptyset$;
3. $G \cap A_0 \neq \emptyset$, $G \cap A_1 = \emptyset$, $(G \cap (A_x \setminus L)) \neq \emptyset$, $(G \cap (A_0 \setminus (L \cap S))) \neq \emptyset$;
4. $G \cap A_1 \neq \emptyset$, $G \cap A_0 = \emptyset$, $(G \cap (A_1 \setminus L)) \neq \emptyset$;
5. $G \cap A_1 \neq \emptyset$, $G \cap A_0 = \emptyset$, $(G \cap (A_x \setminus L)) \neq \emptyset$, $(G \cap (A_1 \setminus (L \cap S))) \neq \emptyset$;
6. $(G \cap (\overline{M \cup L \cup T_0 \cup T_1})) \neq \emptyset$,

выполнена оценка $S(G) \leq 3$.

Перейдем к случаю $P_k, k \geq 3$.

В $P_k, k \geq 3$, вообще говоря, не получен метод определения конечности слоистости произвольной полной системы. Однако произведено сведение проблемы конечности слоистости произвольных полных систем к проблеме конечности слоистости систем Слупецкого.

*Определение* 4. Система Слупецкого в $P_k, k \geq 3$, - это система, состоящая из всех функций, зависящих не более, чем от одной переменной, и существенной функции, принимающей все $k$ значений.

По теореме Слупецкого [3], любая система Слупецкого в $P_k, k \geq 3$, является полной.

**Теорема 2.** *Пусть $G$ - полная система в $P_k, k \geq 3$. Если в $G$ есть существенная функция $g(x_1, \ldots, x_n)$, принимающая все $k$ значений, такая, что соответствующая ей система Слупецкого имеет конечную слоистость, то слоистость системы $G$ конечна.*

Следующая теорема показывает, что для любого $k$ в $P_k, k \geq 3$ существует достаточно широкий класс существенных функций, принимающих все $k$ значений, таких, что соответствующие им системы Слупецкого имеют конечную слоистость.

**Теорема 3.** *Пусть $G$ - система Слупецкого в $P_k$,$k \geq 3$, $g(x_1, \ldots, x_n) \in G$ - существенная функция, принимающая $k$ значений. Пусть существуют номера $i$ и $j$, $i < j$, $i, j \in [1, n]$, а так же набор $(\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_{j-1}, \alpha_{j+1}, \ldots, \alpha_n)$, $\alpha_s \in Z_k$, такой, что $g(\alpha_1, \ldots, \alpha_{i-1}, x_i, \alpha_{i+1}, \ldots, \alpha_{j-1}, x_j, \alpha_{j+1}, \ldots, \alpha_n) = \max(x_i, x_j)$ при $x_i, x_j \in \{0, 1\}$. Тогда система $G$ имеет конечную слоистость.*

Автор выражает благодарность научному руководителю к.ф.-м.н. А.А. Часовских.

**Литература**

1. О.М. Касим-Заде. О глубине булевых функций над произвольным бесконечным базисом // Дискретный анализ и исследование операций. Сер. 1. 2007. 14, №1. 45-69.

2. А.В. Кочергин. О глубине функций k-значной логики в бесконечных базисах // Вестн. Моск. ун-та. Матем. Механ. 2011. №1. 22-26.

3. С.В. Яблонский. Введение в дискретную математику - М.: Высшая школа, 2003.

# ОДНОВРЕМЕННАЯ МИНИМИЗАЦИЯ ОБЪЕМА И МОЩНОСТИ КОНТАКТНЫХ СХЕМ

**Шуткин Ю.С. (Москва, МГУ им. М.В. Ломоносова)**
*yurii.shutkin@gmail.com*

Рассматривается проблема синтеза контактных схем, при котором минимизируется классическая объемная сложность схемы, а также мощностная сложность, характеризующая нагреваемость схемы при функционировании. Отдельно обе меры сложности достаточно изучены. Для объемной сложности основные результаты получены Лупановым [1] и его учениками. Имеет место асимптотика объемной сложности. Мощностная сложность была впервые введена и исследована автором настоящей работы в [2]. В работе [3] было получено асимптотически точное решение.

Посылками к исследованию мощностной сложности схем стали современные требования к схемам. Во-первых, объем схемы постепенно отходит на второй план, так как постоянно развивающиеся технологии позволяют умещать в маленьком физическом объеме все более функционально сложные схемы. На первый же план выходит количество энергии, выделяемой схемой при функционировании, так как зачастую реализовать отвод тепла от схемы либо невозможно, либо очень затратно, а повышенный нагрев может вывести из строя и саму схему, и то, что находится рядом с ней. Во-вторых, введенный функционал сложности оказался полезен для решения еще одной важной задачи. В настоящее время практически весь процесс разработки происходит с использованием компьютера. Сначала схема описывается на некотором специальном языке, а потом тестируется путем симуляции. Таким образом, важно уметь достаточно быстро симулировать различные схемы, что получается далеко не всегда. Оказалось, что введенный функционал сложности позволяет получать оценки и для скорости симуляции схемы, что делает его еще более интересным для исследования.

В настоящей работе исследуется задача одновременной минимизации объемной и мощностной сложности контактных схем. Приведен метод синтеза, асимптотически оптимальный одновременно для обоих функционалов сложности.

**Постановка задачи и формулировка результатов.**

*Контактной схемой* называется неориентированная сеть (т. е.

связный неориентированный граф с двумя выделенными вершинами — полюсами), каждому ребру которой приписана некоторая булева переменная с отрицанием или без него [1]. Ребро вместе с приписанной ему переменной $x_\sigma$ называется *контактом* переменной $x$, а точнее, *замыкающим контактом*, если $\sigma = 1$, и *размыкающим контактом*, если $\sigma = 0$. Контакт $x^\sigma$ считается *замкнутым*, когда $x^\sigma = 1$, и *разомкнутым*, когда $x^\sigma = 0$. Если контакт $x_i^\sigma$ замкнут на наборе $\alpha = (a_1, \ldots, a_n)$, т. е. $a_i^\sigma = 1$, то говорим, что он *проводит* набор $\alpha$, а набор $\alpha$ *проходит* через этот контакт.

По определению, цепь контактной схемы *проводит* набор $\alpha$ тогда и только тогда, когда все ее контакты проводят этот набор.

Контактной схеме ставится в соответствие булева функция (проводимость между полюсами) определенная на наборах значений всех переменных, приписанных контактам схемы, и равная 1 в точности на тех наборах, которые проводятся хотя бы одной цепью между ее полюсами. Говорят также, что схема реализует сопоставленную ей функцию.

Объемной сложностью контактной схемы $S$ называется число контактов в этой схеме. Обозначаем $Q(S)$.

Объемной сложностью реализации функции $f$ контактными схемами $Q(f)$ назовем сложность минимальной схемы, реализующей эту функцию.

$$Q(f) = \min_{S \in H(f)} Q(S),$$

где $H(f)$ — множество всех контактных схем, реализующих функцию $f$.

Для произвольного класса булевых функций $A$ через $A^{(n)}$ будем обозначать класс функций $f \in A$, зависящих от $n$ переменных. Так, $P_2^{(n)}$ будет обозначать класс всех $n$-местных булевых функций.

Функцией Шеннона объемной сложности реализации булевых функций контактными схемами назовем величину

$$Q(n) = \max_{f \in P_2^{(n)}} Q(f).$$

Предположим, что контактная схема $S$ допускает некоторую ориентацию ребер от одного полюса (входного) к другому (выходному) таким образом, что значение реализуемой функции при этом не изменяется.

Для каждого входного набора $\alpha$ рассмотрим множество вершин, для которых проводимость от полюса до вершины с учетом ориентации ребер на этом наборе равна 1. Количество ребер, выходящих из этих вершин, обозначим через $T(S, \alpha)$.

Количество ребер, достижимых на наборе $\alpha$ в ориентированной схеме, называется мощностной сложностью схемы на этом наборе.

Пусть на множестве наборов введено вероятностное пространство $(\{0,1\}^n, \sigma, P)$, где $\sigma$ — множество всех подмножеств булева куба $\{0,1\}^n$, а $P$ — вероятностная мера на этом множестве. Мощностной сложностью контактной схемы назовем величину

$$T(G) = \sum_{\alpha \in \{0,1\}^n} T(G, \alpha) P(\alpha) = E_\alpha\big(T(G, \alpha)\big),$$

где $P(\alpha)$ — вероятность набора $\alpha$ в данном вероятностном пространстве.

Далее будем по умолчанию считать, что распределение наборов равномерное, то есть вероятности появления всех наборов равны.

Мощностной сложностью б. ф. $f$ называется нижняя грань мощностной сложности контактных схем, реализующих эту функцию.

$$T(f) = \inf_{G \in H(f)} T(G).$$

В [2] было показано, что нижняя грань в определении сложности достигается, то есть в $H(f)$ существует контактная схема, сложность которой в точности равна сложности функции $f$. Такие схемы будем называть *оптимальными* для функции $f$, и говорить, что они реализуют функцию $f$ оптимально.

Функцией Шеннона мощностной сложности реализации булевых функций $n$ переменных контактными схемами назовем максимальную сложность функций из $P_2^{(n)}$ (обозначаем $T(n)$).

$$T(n) = \max_{f \in P_2^{(n)}} T(f).$$

В работе [3] были получены следующие оценки.

**Теорема.**

$$T(n) = 2n - 1.$$

**Теорема.** *Для почти всех булевых функций* $f(x_1, \ldots, x_n) \in P_2$ *выполнено*

$$T(f) \sim 2n, \ n \to \infty.$$

В настоящей работе предлагается метод синтеза контактных схем, являющийся асимптотически оптимальным одновременно по объему и по времени.

**Теорема.** *Существует метод синтеза, который каждой булевой функции* $f$ *ставит в соответствие схему* $S(f)$*, реализующую данную функцию, причем для почти всех* $f(x_1, \ldots, x_n) \in P_2$ *выполнено*

$$Q(S(f)) \sim \frac{2^n}{n}, \ n \to \infty,$$

$$T(S(f)) \sim 2n, \ n \to \infty.$$

Автор выражает благодарность своему научному руководителю, проф. Гасанову Эльяру Эльдаровичу за постановку задачи и внимание к работе.

**Литература**

1. Лупанов О.Б. Асимптотические оценки сложности управляющих систем. М.: Изд-во МГУ, 1984.

2. Шуткин Ю.С. О реализации булевых функций информационными графами. // Дискретная математика, 2008, 20:4, 31-41.

3. Шуткин Ю.С. Асимптотически оптимальная реализация булевых функций информационными графами. // Дискретная математика, в печати.

## ON FIVE TYPES OF STABILITY OF MULTICRITERIA COMBINATORIAL MINIMIN PROBLEM

**Emelichev V.A. (Belarusian State University), Karelkina O.V. (University of Turku), Kuzmin K.G. (Belarusian State University)**

*emelichev@bsu.by, volkar@utu.fi, kuzminkg@mail.ru*

In this work, we address the issue of qualitative characteristics of stability of discrete multicriteria optimization problems (see, e.g. [2]). Analysis of the five most known stability types has been carried out for two multicriteria minimin combinatorial problems: with Pareto and lexicographic principles of optimality. As a result necessary and at the same time sufficient conditions for each stability type are obtained as well as interrelation between these types are revealed.

Let $A_i$ be the $i$-th row of matrix $A = [a_{ij}] \in \mathbf{R}^{n\times m}$, $n \geq 1$, $m \geq 2$, $T$ be a non empty system of non empty sets $N_m = \{1, 2, \ldots, m\}$ (called trajectories), i.e. $T \subseteq 2^{N_m} \setminus \{\emptyset\}$, $|T| \geq 2$. Let the components of vector-function $f(t, A) = (f_1(t, A_1), f_2(t, A_2), \ldots, f_n(t, A_n))$ be defined on $T$ by minimin criteria (see, e.g. [3])

$$f_i(t, A_i) = \min_{j\in t} a_{ij} \to \min_{t\in T}, \quad i \in N_n.$$

On the set $T$ we define two binary relations of domination

$$t \underset{P,A}{\succ} t' \;\Leftrightarrow\; f(t, A) \geq f(t', A) \;\wedge\; f(t, A) \neq f(t', A),$$

$$t \underset{L,A}{\succ} t' \;\Leftrightarrow\; \exists k \in N_n \;\big(f_k(t, A_k) > f_k(t', A_k) \wedge$$

$$\wedge\; k = \min\{i \in N_n : f_i(t, A_i) \neq f_i(t', A_i)\}\big).$$

Using these relations we specify the Pareto set and lexicographic set respectively:

$$P^n(A) = \big\{t \in T :\; \forall t' \in T \;\; (t \underset{P,A}{\not\succ} t')\big\},$$

$$L^n(A) = \big\{t \in T :\; \forall t' \in T \;\; (t \underset{L,A}{\not\succ} t')\big\}.$$

Here and further the line over a binary relation means the negation of the relation.

Thus two $n$-criteria combinatorial problems with minimin criteria arise: the problem $Z_P^n(A)$ of finding the Pareto set $P^n(A)$ and the problem $Z_L^n(A)$ of finding the lexicographic set $L^n(A)$.

Since $2 \le |T| < \infty$ then $\emptyset \ne L^n(A) \subseteq P^n(A)$ for any $A \in \mathbf{R}^{n\times m}$.

Let us put into consideration the Smale set and the Slater set respectively:

$$Sm^n(A) = \{t \in T : \ \forall t' \in T \setminus \{t\} \ \ (t \underset{Sm,A}{\succ} t')\},$$

$$Sl^n(A) = \{t \in T : \ \forall t' \in T \setminus \{t\} \ \ (t \underset{Sl,A}{\succ} t')\},$$

where $t \underset{Sm,A}{\succ} t' \ \Leftrightarrow \ f(t,A) \ge f(t',A)$ and $t \underset{Sl,A}{\succ} t' \ \Leftrightarrow \ f(t,A) > f(t',A)$.

Let us denote for brevity any of the sets $P^n(A)$ or $L^n(A)$ by $M^n(A)$ and a multicriteria problem of finding $M^n(A)$ by $Z^n_M(A)$.

We will investigate the five known (see, e.g., [1]) stability types of the multicriteria problem $Z^n_M(A)$. The problem $Z^n_M(A)$ is called $S_1$-*stable* if there exists $\varepsilon > 0$ such that for any $A' \in \Omega(\varepsilon)$ we have $M^n(A + A') \subseteq M^n(A)$; $S_2$-*stable* if there exists $\varepsilon > 0$ such that for any $A' \in \Omega(\varepsilon)$ we have $M^n(A) \cap M^n(A + A') \ne \emptyset$; $S_3$-*stable* if there exists $\varepsilon > 0$ such that for any $A' \in \Omega(\varepsilon)$ we have $M^n(A) \subseteq M^n(A + A')$; $S_4$-*stable* if there exists $\varepsilon > 0$ such that for any $A' \in \Omega(\varepsilon)$ we have $M^n(A) = M^n(A + A')$ and $S_5$-*stable* if there exist $\varepsilon > 0$ and $t^0 \in M^n(A)$ such that for any $A' \in \Omega(\varepsilon)$ we have $t^0 \in M^n(A + A')$. Here $\Omega(\varepsilon) = \{A' \in \mathbf{R}^{n\times m} : \ \|A'\| < \varepsilon\}$ is the set of perturbing matrices $A' = [a'_{ij}]$ with rows $A'_i$, $i \in N_n$, $\|A'\| = \max\{|a'_{ij}| : \ (i,j) \in N_n \times N_m\}$.

**Remark 1.** *Directly from the given definitions it follows:*
*1) if the problem $Z^n_M(A)$ is $S_1$-stable, then it is $S_2$-stable,*
*2) if the problem $Z^n_M(A)$ is $S_3$-stable, then it is $S_5$-stable,*
*3) the problem $Z^n_M(A)$ is $S_4$-stable if and only if it is $S_1$- and $S_3$-stable,*
*4) if the problem $Z^n_M(A)$ is $S_5$-stable, then it is $S_2$-stable.*

Let us denote $N_i(t, A_i) = \operatorname{Arg\,min}\{a_{ij} : j \in t\}$, $i \in N_n$, and $V(t, A, I) = \prod\limits_{i\in I} N_i(t, A_i)$, $I \subseteq N_n$.

**The problem with Pareto principle of optimality**

For vector $v = (v_1, v_2, \ldots, v_n) \in \mathbf{R}^n$ and set $I = \{i_1, i_2, \ldots, i_k\} \subseteq N_n$, $i_1 < i_2 < \ldots < i_k$, we introduce notation $v_I = (v_{i_1}, v_{i_2}, \ldots, v_{i_k})$.

We put $P^n(t, A) = \{t' \in P^n(A) : \ f(t, A) \ge f(t', A)\}$,

$$I(t, t') = \{i \in N_n : \ f_i(t, A_i) = f_i(t', A_i)\}.$$

**Theorem 1.** *$Z_P^n(A)$, $n \geq 1$, is $S_1$-stable iff for any trajectory $t \in Sl^n(A)$ and vector $v \in V(t, A, N_n)$ there exists trajectory $t^* \in P^n(t, A)$ such that $v_{I(t,t^*)} \in V(t^*, A, I(t, t^*)$.*

This statement indicates that for any trajectory $t \in Sl^n(A)$ there exists trajectory $t^* \in P^n(t, A)$ which is invariant to small perturbations of problem parameters.

**Theorem 2.** *$Z_P^n(A)$, $n \geq 1$, is $S_2$-stable for any matrix $A \in \mathbf{R}^{n \times m}$.*

For trajectory $t \in P^n(A)$ we introduce a set $Q(t, A) = \{t' \in T : f(t, A) = f(t', A)\}$.

**Theorem 3.** *$Z_P^n(A)$, $n \geq 1$, is $S_3$-stable iff for any trajectories $t \in P^n(A)$, $t' \in Q(t, A)$ and any index $i \in N_n$ the inclusion $N_i(t, A_i) \supseteq N_i(t', A_i)$ is valid.*

Condition given above indicates that for any two equivalent trajectories $t$ and $t'$ the equality $V(t, A, N_n) = V(t', A, N_n)$ must hold.

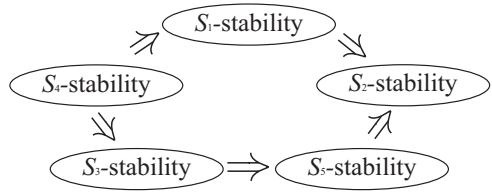
The next result follows from theorems 1 and 3 by virtue of remark 1.

**Theorem 4.** *$Z_P^n(A)$, $n \geq 1$, is $S_4$-stable iff both statements hold:*
(i) $\forall t \in Sl^n(A) \quad \forall v \in V(t, A, N_n) \quad \exists t^* \in P^n(t, A) \quad \big(v \in V(t^*, A, I(t, t^*))\big)$,
(ii) $\forall t \in P^n(A) \quad \forall t' \in Q(t, A) \quad \forall i \in N_n \quad \big(N_i(t, A_i) \supseteq N_i(t', A_i)\big)$.

**Theorem 5.** *$Z_P^n(A)$, $n \geq 1$, is $S_5$-stable iff there exists trajectory $t^0 \in P^n(A)$ such that for any trajectory $t \in Q(t^0, A)$ and any index $i \in N_n$ the inclusion $N_i(t^0, A_i) \supseteq N_i(t, A_i)$ is valid.*

This condition indicates of the existence of efficient trajectory $t^0$ such that for all trajectories $t$ equivalent to it the inclusion $V(t^0, A, N_n) \supseteq V(t, A, N_n)$ holds.

Summarizing the results obtained in theorems 1–5 and taking into account remark 1, we conclude that relations between different stability types of the problem $Z_P^n(A)$ are described by the following scheme:




**The problem with lexicographic principle of optimality**

Let us introduce a set of indexes $M(t) = \{i \in N_n : \ t \in L_i^n(A_i)\}$. It is easy to see that for $t \in L_1^n(A)$ we have $\emptyset \neq M(t) = N_q \subseteq N_n$, where $q = \max\{i \in N_n : t \in L_i^n(A)\} = |M(t)|$.

**Theorem 6.** *For the problem $Z_L^n(A)$, $n \geq 1$, the following statements are equivalent:*
(i) *$Z_L^n(A)$ is $S_1$-stable,* (ii) *$Z_L^n(A)$ is $S_2$-stable,*
(iii) $\forall t \in L_1^n(A) \quad \forall v \in V(t, A, M(t)) \quad \exists t^* \in L^n(A) \quad \big(v \in V(t^*, A, M(t))\big)$.

Statement (iii) indicates that for any non lexicographic trajectory $t \in L_1^n(A)$ there exists trajectory $t^* \in L^n(A)$ that will not allow trajectory $t$ to become lexicographically optimal under small perturbations.

**Theorem 7.** *For the problem $Z_L^n(A)$, $n \geq 1$, the following statements are equivalent:*
(i) *$Z_L^n(A)$ is $S_3$-stable,* (ii) *$Z_L^n(A)$ is $S_4$-stable,*
(iii) $\forall t \in L^n(A) \ \ \forall i \in N_n \ \ \forall t' \in L_i^n(A) \quad \big(N_i(t, A_i) \supseteq N_i(t', A_i)\big)$.

Statement (iii) indicates that any trajectory $t \in L^n(A)$ must not be dominated by trajectories $L_i^n(A)$, $i \in N_n$, under small perturbations of problem parameters.

**Theorem 8.** *$Z_L^n(A)$, $n \geq 1$, is $S_5$-stable iff there exists trajectory $t^0 \in L^n(A)$ such that for any index $i \in N_n$ and any trajectory $t \in L_i^n(A)$ the inclusion $N_i(t^0, A_i) \supseteq N_i(t, A_i)$ is valid.*

This statement indicates of that there exists lexicographically optimal trajectory $t^0$ which must not be dominated by trajectories $L_i^n(A)$, $i \in N_n$, under small perturbations of problem parameters.

Summarizing the results obtained in theorems 6, 7 and 8, taking into account remark 1, we conclude that the relations between different stability types of the problem $Z_L^n(A)$ are described by the following scheme:


## References

1. Emelichev V.A., Girlich E., Nikulin Yu.V. and Podkopaev D.P. Stability and regularization of vector problem of integer linear programming // Optimization **51** (4), 2002, pp. 645-676.

2. Sergienko I.V. and Shilo V.P. // Discrete Optimization Problems (in Russian). – Kiev: Naukova dumka, 2003.

3. Tapiero Ch. // Risk and Financial Management. – Chichester: John Wiley and Sons Ltd, 2004.

## ON THE 2D ORDER CURVES OVER FINITE RING

**Skobelev V.V. (Ukraine, Donetsk, IAMM of NAS of Ukraine)**
*vv_ skobelev@iamm.ac.donetsk.ua*

Applications of algebraic models (especially of elliptic curves) in modern cryptography has stimulated research of properties of algebraic curves over arbitrary finite associative-commutative ring $\mathcal{K} = (K, +, \cdot)$ with the unit. In [1] systematically investigated the 2d and the 3d order curves. In the given paper are presented some properties of curves

$$\Gamma : a_{11}x^2 + a_{12}xy + a_{22}y^2 + a_1x + a_2y + a_0 = 0, \qquad (1)$$

where $a_{11}, a_{12}, a_{22}, a_1, a_2, a_0 \in K$ and $(a_{11}, a_{12}, a_{22}) \neq (0, 0, 0)$.

The graph of $\Gamma$ can be characterized in the following way.

Let $a_{3-i,3-i} \neq 0$ and $a_{ii} = a_{12} = a_i = 0$ where either $i = 1$, or $i = 2$. The graph of $\Gamma$ is the set of all solutions $(x_0, y_0) \in K^2$ either of equation $a_{22}y^2 + a_2y + a_0 = 0$ $(i = 1)$, or of equation $a_{11}x^2 + a_1x + a_0 = 0$ $(i = 2)$.

Let $a_{3-i,3-i} \neq 0$, $a_{ii} = a_{12} = 0$ and $a_i \neq 0$ where either $i = 1$, or $i = 2$.

**Theorem 1.** *Let Char* $\mathcal{K} \neq 2$. *If* $i = 1$ *and there exist elements* $b, c \in K\backslash\{0\}$ *such that* $a_{22} = cb^2$ *and* $a_2 = 2cb$ *then the graph of* $\Gamma$ *is the set of all points* $(x_0, y_0) \in K^2$ *such that* $(w_0, u_0) = (by_0 + 1, x_0)$ *is a solution of equation* $cw^2 + a_1u + (a_0 - c) = 0$. *If* $i = 2$ *and there exist elements* $b, c \in K\backslash\{0\}$ *such that* $a_{11} = cb^2$ *u* $a_1 = 2cb$ *then the graph of* $\Gamma$ *is the set of all points* $(x_0, y_0) \in K^2$ *such that* $(w_0, u_0) = (bx_0 + 1, y_0)$ *is a solution of equation* $cw^2 + a_1u + (a_0 - c) = 0$.

Let $a_{11} = a_{22} = 0$ and $a_{12} \neq 0$.

**Theorem 2.** *If* $a_1 = a_2 = 0$ *then the graph of* $\Gamma$ *is the set of all points* $(x_0, y_0) \in K^2$ *such that* $(x_0, y_0)$ *is a solution of equation* $a_{12}xy + a_0 = 0$. *If* $a_2 \neq 0$ *and there exists an element* $c \in K\backslash\{0\}$ *such that* $a_2 = ca_{12}$ *then the graph of* $\Gamma$ *is the set of all points* $(x_0, y_0) \in K^2$ *such that* $(u_0, v_0) = (a_{12}y_0 + a_1, x_0 + c)$ *is a solution of equation* $uv + (a_0 - ca_1) = 0$. *If* $a_1 \neq 0$ *and there exists an element* $c \in K\backslash\{0\}$ *such that* $a_1 = ca_{12}$ *then the graph of* $\Gamma$ *is the set of all points* $(x_0, y_0) \in K^2$ *such that* $(u_0, v_0) = (a_{12}x_0 + a_2, y_0 + c)$ *is a solution of equation* $uv + (a_0 - ca_2) = 0$.

Let $a_{ii} \neq 0$ $(i = 1, 2)$ and $a_j \neq 0$ $(j = 1, 2)$.

**Theorem 3.** Let *Char* $\mathcal{K} \neq 2$ and there exist elements $b_1, b_2, c, d \in K\backslash\{0\}$ such that $a_{11} = cb_1^2$, $a_{12} = 2cb_1b_2$, $a_{22} = cb_2^2$, $a_1 = db_1$ and

$a_2 = db_2$. The graph of $\Gamma$ is the set of all points $(x_0, y_0) \in K^2$ such that $w_0 = b_1x_0 + b_2y_0$ is a solution of equation $cw^2 + dw + a_0 = 0$.

Let $a_{ii} \neq 0$, $a_{3-i,3-i} = 0$ and $a_{12} \neq 0$, where either $i = 1$, or $i = 2$. The graph of $\Gamma$ is the set of all solutions $(x_0, y_0) \in K^2$ either of equation $x(a_{11}x + a_{12}y) + a_1x + a_2y + a_0 = 0$ $(i = 1)$, or of equation $y(a_{22}y + a_{12}x) + a_1x + a_2y + a_0 = 0$ $(i = 2)$.

Let $Char\ \mathcal{K} \neq 2$, $a_{ii} \neq 0$ $(i = 1, 2)$ and either $a_1 = 0$, or $a_2 = 0$. If $a_1 = 0$ and there exist elements $b_1, b_2, c, d \in K\backslash\{0\}$ such that $a_{22} = db^2$, $a_2 = 2dbc$ and $a_0 = dc^2$ then the graph of $\Gamma$ is the set of all solutions $(x_0, y_0) \in K^2$ of equation $x(a_{11}x + a_{12}y) + d(by + c)^2 = 0$. If $a_2 = 0$ and there exist elements $b_1, b_2, c, d \in K\backslash\{0\}$ such that $a_{11} = db^2$, $a_1 = 2dbc$ and $a_0 = dc^2$ then the graph of $\Gamma$ is the set of all solutions $(x_0, y_0) \in K^2$ of equation $y(a_{22}y + a_{12}x) + d(bx + c)^2 = 0$.

Canonical form of $\Gamma$ can be characterized in the following way.

Let

$$\begin{pmatrix} x \\ y \end{pmatrix} = A \begin{pmatrix} u \\ v \end{pmatrix},$$

where

$$A = \begin{pmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{pmatrix}.$$

Linear operator $A$ annihilates linear form

$$h(x, y) = a_1x + a_2y$$

if and only if $a_1\alpha_{11} + a_2\alpha_{21} = 0$ and $a_1\alpha_{12} + a_2\alpha_{22} = 0$. Thus any linear operator that annihilates linear form $h(x, y)$ is not a bijection.

**Theorem 4.** *In the result of linear transformation $A$ quadric form*

$$f(x, y) = a_{11}x^2 + a_{12}xy + a_{22}y^2$$

*can be transformed into the form $g(u, v) = b_{11}u^2 + b_{22}v^2$ if and only if there exist elements $\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22} \in K$, such that*

$$2(a_{11}\alpha_{11}\alpha_{12} + a_{22}\alpha_{21}\alpha_{22}) + a_{12}(\alpha_{11}\alpha_{22} + \alpha_{12}\alpha_{21}) = 0,$$

*where*

$$b_{11} = a_{11}\alpha_{11}^2 + a_{12}\alpha_{11}\alpha_{21} + a_{22}\alpha_{21}^2$$

*and*

$$b_{22} = a_{11}\alpha_{12}^2 + a_{12}\alpha_{12}\alpha_{22} + a_{22}\alpha_{22}^2.$$

**Corollary 1.** *In the result of linear transformation A quadric form* $f(x,y)$ *can be transformed into the form* $g(u,v) = b_{11}u^2$ *if and only if there exist elements* $\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22} \in K$ *such that*

$$2(a_{11}\alpha_{11}\alpha_{12} + a_{22}\alpha_{21}\alpha_{22}) + a_{12}(\alpha_{11}\alpha_{22} + \alpha_{12}\alpha_{21}) = 0$$

*and*

$$a_{11}\alpha_{12}^2 + a_{12}\alpha_{12}\alpha_{22} + a_{22}\alpha_{22}^2 = 0.$$

**Corollary 2.** *In the result of linear transformation A quadric form* $f(x,y)$ *can be transformed into the form* $g(u,v) = b_{22}v^2$ *if and only if there exist elements* $\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22} \in K$ *such that*

$$2(a_{11}\alpha_{11}\alpha_{12} + a_{22}\alpha_{21}\alpha_{22}) + a_{12}(\alpha_{11}\alpha_{22} + \alpha_{12}\alpha_{21}) = 0$$

*and*

$$a_{11}\alpha_{11}^2 + a_{12}\alpha_{11}\alpha_{21} + a_{22}\alpha_{21}^2 = 0.$$

**Corollary 3.** *In the result of linear transformation A quadric form* $f(x,y)$ *can be transformed into the form* $g(u,v) = b_{12}uv$ *if and only if there exist elements* $\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22} \in K$ *such that*

$$a_{11}\alpha_{11}^2 + a_{12}\alpha_{11}\alpha_{21} + a_{22}\alpha_{21}^2 = 0$$

*and*

$$a_{11}\alpha_{12}^2 + a_{12}\alpha_{12}\alpha_{22} + a_{22}\alpha_{22}^2 = 0.$$

**Remark.** *The reason for extraction of equation* $b_{12}uv + a_0 = 0$ *is the following. Linear form* $b_{12}uv + a_0$ *in a ring can be transformed by bijection* $u = \gamma U + \delta V$, $v = \varphi U + \psi V$ *into the form* $\gamma\varphi U^2 + \delta\psi V^2$ *if and only if there exists elements* $\gamma, \delta, \varphi, \psi \in K$ *such that , что* $\gamma\psi - \delta\varphi \neq 0$ *and* $b_{12}(\gamma\psi + \delta\varphi) = 0$.

Further investigations can be connected with analysis of non-trivial subclasses of curves.

## References

1. V.V. Skobelev, N.M. Glazunov, V.G. Skobelev. Manifolds over rings. Theory and applications. — Donetsk: IAMM of NAS of Ukraine, 2011. – 323 p.

# Секция “Информатика и прикладные исследования”

# ОПТИМАЛЬНЫЙ МЕТОД ПЕРЕРАСПРЕДЕЛЕНИЯ ОБЩЕЙ ПАМЯТИ ДЛЯ ДВУХПРИОРИТЕТНОЙ ОЧЕРЕДИ, ПРЕДСТАВЛЕННОЙ В ВИДЕ ДВУХ ПОСЛЕДОВАТЕЛЬНЫХ ЦИКЛИЧЕСКИХ FIFO-ОЧЕРЕДЕЙ

**Аксенова Е.А., Соколов А.В. (Петрозаводск, Институт прикладных математических исследований КарНЦ РАН, Петрозаводский государственный университет)**

*aksenova@krc.kareia.ru, avs@krc.karelia.ru*

Во многих приложениях используется структура данных, в которой основными операциями являются вставка элемента и удаление элемента с наибольшим приоритетом. Такую структуру данных называют приоритетной очередью. Основными методами реализации приоритетной очереди являются упорядоченные и неупорядоченные списки, массивы, бинарные деревья, пирамиды [1]-[3]. Эти методы сравниваются с точки зрения стоимости (времени) выполнения основных операций. В работах [4]-[7] предлагались математические модели для последовательного циклического и связанного способов представления очередей и решались задачи оптимального начального распределения памяти для разных критериев оптимальности.

Целью данной работы является изучение динамики изменения длины очереди в случае представления $n$-приоритетной очереди в виде $n$ последовательных FIFO-очередей, когда все элементы с одинаковыми приоритетами помещаются в одну FIFO-очередь. Каждой FIFO-очереди выделен свой участок памяти, поэтому нет затрат памяти на хранение приоритетов. Одна из FIFO-очередей может исчерпать свою память, а у остальных очередей в этот момент еще будет свободная память. При этом допускается возможность переполнения FIFO-очереди, что, например, в маршрутизаторах является обычной ситуацией. В данной работе предлагается математическая модель и алгоритм оптимального перераспределения памяти после переполнения одной из FIFO-очередей. В качестве критерия оптимальности рассмотрено максимальное среднее время работы до переполнения.

Рассмотрим очередь с двумя приоритетами, расположенную памяти размера $m$ единиц. Будем считать, что время дискретно, и в каждый момент времени происходит одна из следующих операций:
– вставить элемент с приоритетом $i$ с вероятностью $p_i$ $(i = 1, 2)$,
– удалить элемент с наибольшим приоритетом с вероятностью $q$,

– найти элемент с наибольшим приоритетом с вероятностью $r$, где $p_1 + p_2 + q + r = 1$.

Вероятности не зависят от времени и количества элементов в очереди. Наивысший приоритет равен 2, наименьший – 1. Работа начинается с пустой очереди и не завершается при исключении элемента из пустой очереди. Исключение элемента из очереди происходит по наивысшему приоритету. Это означает, что пока вторая FIFO-очередь не пуста, с вероятностью $q$ исключение элементов происходит из этой очереди. Как только вторая FIFO-очередь станет пустой, с вероятностью $q$ исключение элементов будет происходить из первой FIFO-очереди. Предполагаем, что при переполнении одной из FIFO-очередей происходит перераспределение свободной памяти между очередями и работа с программной системой продолжается.

Выделим первой FIFO-очереди $s$ единиц памяти, тогда для второй останется $m - s$ единиц. Обозначим текущие длины очередей $x_1$ и $x_2$. В качестве математической модели рассмотрим случайное блуждание по целочисленной решетке в прямоугольной области на плоскости $0 \le x_1 < s+1$, $0 \le x_2 < m-s+1$. Блуждание начинается в точке $x_1 = 0, x_2 = 0$. Переполнению первой FIFO-очереди соответствуют точки на прямой $x_1 = s+1$, переполнению второй - точки на прямой $x_2 = m - s + 1$. Прямые $x_1 = -1$ и $x_2 = -1$ - отражающие экраны, т.е. при исключении элемента из пустой очереди работа не завершается.

Требуется оптимально перераспределить свободную память между FIFO-очередями после переполнения одной из них. То есть, при попытке включения элемента в заполненную FIFO-очередь, когда $x_1 = s$ (или $x_2 = m - s$), требуется определить новую область блуждания $0 \le x_1 < s^* + 1$, $0 \le x_2 < m - s^* + 1$ , т.е. такое значение $s^*$, где $s^* > s$ ( или $m - s^* > m - s$), чтобы время до следующего переполнения какой-либо FIFO-очереди было максимальным. Этот процесс перераспределения можно продолжать до полного исчерпания свободной памяти. Такой подход оправдан, если при переполнении одной из FIFO-очередей, в области памяти, выделенной для других очередей, есть достаточно свободной памяти. В алгоритме Гарвика [1] для работы с $n$ последовательными стеками и очередями при переполнении какой-либо структуры данных приблизительно 10% свободной памяти делится поровну между структурами данных, а оставшиеся 90% делятся пропорционально росту размеров структур данных с

момента предыдущего распределения памяти.

Математическая постановка задачи сводится к решению задачи нелинейного целочисленного программирования, с критерием оптимальности, заданным алгоритмически. Случайное блуждание по целочисленной решетке будем рассматривать как конечную однородную поглощающую цепь Маркова с матрицей вероятностей переходов P [8]. Для решения задачи требуется матрица $Q$ вероятностей переходов из невозвратных состояний в невозвратные ($Q$ - подматрица матрицы $P$). Вводится нумерация состояний Марковской цепи, это позволяет определить структуру матрицы $Q$ для любого размера памяти $m$ и любого значения $s$. Среднее время работы до переполнения будем искать с помощью фундаментальной матрицы $N = (E-Q)^{-1}$. Предполагаем, что в самом начале работы, когда обе очереди пустые $x_1 = 0, x_2 = 0$, память между ними тоже нужно разделить оптимально, т.е. выбрать такое $s$, чтобы время работы с очередями до переполнения было максимально.

**Алгоритм:**

1. Ввод вероятностных характеристик приоритетной очереди, размера памяти $m$;
2. Для каждого значения $s, 0 \leq s \leq m$, генерируем матрицу $Q$, вычисляем матрицу $N$;
3. Для каждого значения $s$ суммируем элементы матрицы $N$ в строке, соответствующей состоянию $x_1 = 0, x_2 = 0$;
4. Выбираем такое значение $s$, которому соответствует максимальная сумма элементов в строке соответствующей матрицы $N$;
5. Для каждого состояния, в котором одна из FIFO-очередей заполнила выделенную память, т. е. $x_1 = s$ или $x_2 = m - s$, перебираем значения $0 \leq s^* \leq m$, где $s^* > s$, если $x_1 = s$, и $m - s^* > m - s$, если $x_2 = m - s$. Для каждого $s^*$ генерируем новую матрицу $Q$, вычисляем матрицу $N$;
6. Для каждого значения $s^*$ суммируем элементы матрицы $N$ в строке, соответствующей рассматриваемому состоянию ( $x_1 = s$ или $x_2 = m - s$);
7. Выбираем такое значение $s^*$, которому соответствует максимальная сумма элементов в строке соответствующей матрицы $N$. Для каждого состояния, соответствующего тому, что одна из очередей заполнила выделенную память, получаем оптимальное значение $s^*$.

Шаги 5-7 можно повторять до полного исчерпания свободной па-

мяти (если перед каждым новым прераспределением текущее значение $s^*$ обозначить как $s$).

Вычисление критерия оптимальности в данной задаче является очень ресурсоемким за счет обращения матрицы $(E - Q)$ большого размера. Матрицу $N$ можно представить в виде $N = (E - Q)^{-1} = E + Q + Q^2 + ... = \sum_{k=0}^{\infty} Q^k$. Для данной задачи такое представление матрицы дает выигрыш в размере памяти для вычислений. Поскольку для алгоритма важна сумма элементов в определенной строке матрицы N, то запись в виде суммы ряда позволяет вычислять элементы конкретной строки, не вычисляя остальных элементов матрицы.

Работа выполнена при финансовой поддержке РФФИ, грант 09-01-00330.

**Литература**

1. Кнут Д. Искусство программирования для ЭВМ. – М.: Вильямс, 2001.

2. Седжвик Р. Фундаментальные алгоритмы на C++. – К.: Диасофт, 2001.

3. Боллапрагада В., Мэрфи К., Уайт Р. Структура операционной системы Cisco IOS. – М.: Вильямс, 2002.

4. Аксенова Е. А., Соколов А. В. Некоторые задачи оптимального управления FIFO-очередями // Труды Второй Всероссийской научной конференции "Методы и средства обработки информации". – М.: Изд. отдел ВМК МГУ им. М.В. Ломоносова, 2005, с. 318-322.

5. Аксенова Е. А. Оптимальное управление FIFO-очередями на бесконечном времени // Межвузовский сборник "Стохастическая оптимизация в информатике". – СПб: Изд-во С.-Петербургского университета, 2006, с. 71-76.

6. Аксенова Е. А., Драц А. В., Соколов А. В. Об оптимальном управлении FIFO-очередями на бесконечном времени // Обозрение прикладной и промышленной математики. – Т.16, вып.3, 2009, с. 401-415.

7. Аксенова Е. А., Драц А. В., Соколов А. В. Оптимальное управление n FIFO-очередями на бесконечном времени // Информационно-управляющие системы. – № 6, 2009, с. 46-54.

8. Кемени Дж., Снелл Дж. Конечные цепи Маркова. – М.: Наука, 1970.

# ПОСТРОЕНИЕ И АНАЛИЗ ДЕТЕРМИНИРОВАННЫХ МЕТОДОВ ПРОГНОЗИРОВАНИЯ

**Андреев А.В. (Москва, МГУ им. М.В. Ломоносова)**
**Пытьев Ю.П. (Москва, МГУ им. М.В. Ломоносова)**
*alvlandr@gmail.com, yuri.pytyev@gmail.com*

В работе рассмотрены математические методы краткосрочного прогнозирования.

Целью данной работы являлось построение и исследование методов, обеспечивающих высокую точность прогноза стоимостей акций на $l$ шагов вперёд («на $l$ следующих дней»).

**Прогнозирование стоимости акций на $l$ «следующих дней».** Сформирован портфель из акций пяти различных компаний. Известна динамика изменения курса этих акций — стоимость акций за определённый период времени: $n$ дней. Требуется оценить стоимость акции каждой компании в период с $(n+1)$-го по $(n+l)$-ый день. В работе представлены три подхода, позволяющие решить поставленную задачу.

**Метод с предиктором.** Данный метод является методом линейного прогнозирования, аналогичным методу наименьших квадратов (МНК). В его основе лежит решение следующей задачи на минимум:

$$||YX - Z|| \sim \min_{X \in R^M} \tag{1}$$

где $Y$ — матрица, каждая строка которой составлена из пяти временных рядов (по количеству типов акций) одинаковой длины, элементами (уровнями) которых являются стоимости акций, следующая строка получается сдвигом предыдущей на один шаг вперёд (таким образом, что в каждой следующей строке кроме первой используется пять новых значений относительно предыдущей), $X$ — предсказывающая матрица или предиктор, $Z$ — матрица, каждая строка которой состоит из $5l$ значений стоимостей акций, опережающих значения в соответствующей строке матрицы $Y$ на число шагов от 1 до $l$. Проще говоря, при умножении одной строки матрицы $Y$ на предиктор появятся $5l$ будущих значений стоимостей акций, составляющих матрицу $Z$.

Вероятностный аналог данного метода рассмотрен в [2].

**Метод, минимизирующий среднеквадратичное отклонение.** Пусть $A_k$ — матрица размера $5\times m$, каждая из пяти строк которой (по числу типов акций) представляет собой временной ряд длины $m$, $\alpha_k$:$5\times l$ — вектор, составленный из цен акций, опережающих соответствующие значения в $A_k$ на один шаг (на один день). Введём векторы $x$:$m\times l$ и $y$:$5\times l$ такие, что преобразование $A_k x + y$ было как можно ближе в среднеквадратичном к $\alpha_k$. Таким образом, необходимо решить следующую задачу:

$$\frac{1}{s-1}\sum_{k=m+1}^{s+m} ||\alpha_k - A_k x - y||^2 \sim \min_{x,y} \qquad (2)$$

Требуется решить задачу на минимум (2) и найти $x$, $y$, если известны вектор $\alpha_k$ и матрица $A_k$.

**Анализ методов.** В результате проведённого численного исследования построенных методов был получен ответ на вопрос: сколько предыдущих значений следует использовать при прогнозе будущего. Данная характеристика соответствует значению величины $m$.Усреднённая относительная ошибка прогноза оказалась наименьшей при $m = 2$. Однако такое значение параметра являлось не равномерно наилучшим, поэтому были разработаны два адаптивных алгоритма выбора $m$. Первый алгоритм вычислял прогноз при нескольких значениях $m$, и то значение, при котором относительная ошибка оказывалась наименьшей, выбиралось для прогноза на следующий период. Во втором алгоритме подсчитывалось, как часто то или иное значение $m$ обеспечивало наименьшую ошибку, и в дальнейшем значения параметра разыгрывались, т.е. применялась рандомизация. Реализация этих алгоритмов показала, что такой адаптивный выбор $m$ не позволил повысить качество прогноза.

**Применение сглаживающих сплайнов в задаче прогнозирования.** Проведённый анализ, показавший, что качество прогноза получается в среднем наилучшим при выборе параметра $m = 2$, позволяет предположить, что для решения поставленной задачи можно использовать сплайны. В частности, при решении задачи интерполяции через точки $(t_1, r_1), \ldots, (t_n, r_n)$ требуется провести в известном

смысле самую гладкую кривую. Можно показать, что решением этой задачи является сплайн. Однако гладкость сплайна, проходящего через заданные $n$ точек, может оказаться недостаточной. В таком случае не следует требовать, чтобы сплайн точно проходил через указанные точки. Это тем более разумно, если точки получены из приблизительных измерений. Компромисс между гладкостью кривой и точностью аппроксимации при $t = t_1, t_2, \ldots, t_n$ можно реализовать, решая задачу на минимизацию:

$$\min\{\int_a^b (f^{(q)})^2(t)\,dt + \rho\sum_{j=1}^{n}(f(t_j) - r_j)^2 : f{\in}H^q\} \tag{3}$$

при должным образом выбранном $\rho > 0$, где $H^q$ — гильбертово пространство функций на $[a, b]$ с абсолютно непрерывной $q-1$ производной и суммируемых в квадрате $q$ производной:

$$\int_a^b (f^{(q)})^2(t)\,dt < \infty \tag{4}$$

Решением этой задачи также является сплайн, называемый сглаживающим, его гладкость тем выше, чем меньше $\rho$.

**Результаты и выводы** Предложены методы прогнозирования изменения динамики курсов акций, обеспечивающие точность предсказания сравнимую с уже известными методами. Для сравнения методов между собой использовалась величина $E^{(k)}$, значение которой для каждого метода характеризует качество прогноза, получаемого при использовании соответствующей метода:

$$E^{(k)} = \frac{1}{N}\sum_{i=1}^{N}\frac{|k_i - k_i^{pr}|}{k_i}, \tag{5}$$

где $k_i$ — стоимость акции в $i$-ый день, $k_i^{pr}$ — полученное в результате прогноза значение стоимости акции в $i$-ый день, $N$ — количество сделанных прогнозов.

Результаты использования описанных выше методов приведены в таблице 1. Помимо разработанных подходов в сравнении участвовали методы Брауна и Хольта. Полученные значения ошибок указывают на то, что все эти методы сопоставимы, и из них нельзя

выделить один равномерно наиболее точный. В тоже время можно заметить, что средняя ошибка прогноза метода с предиктором равномерно выше ошибки метода, минимизирующего среднеквадратичное отклонение. Разница между первым и вторым методами такая же, как между МНК и математической редукцией, и проигрыш метода с предиктором в качестве прогноза связан с тем, что в первом случае мы минимизируем невязку, а во втором случае – ошибку.

| | Сбербанк | 7 Континент | Северсталь | МСРК | Лукойл |
|---|---|---|---|---|---|
| МП | 0.039 | 0.032 | 0.036 | 0.025 | 0.027 |
| ММСК | 0.042 | 0.025 | 0.033 | 0.022 | 0.026 |
| МС | 0.038 | 0.030 | 0.034 | 0.025 | 0.028 |
| МБ | 0.049 | 0.034 | 0.043 | 0.033 | 0.034 |
| МХ | 0.036 | 0.027 | 0.032 | 0.024 | 0.027 |

Таблица 1: *Значения $E^{(k)}$ для каждого метода. Параметр $m = 2$. МП — метод с предиктором, ММСК — метод, минимизирующий с.к. отклонение, МС — метод, использующий сглаживающие сплайны, МБ — метод Брауна, МХ — метод Хольта*

Судя по полученным результатам, не следует ожидать улучшения качества прогноза без учёта модели данных. Если модель известна приближённо, то можно поставить задачу оптимального предсказания. Приведём пример модели данных, в которых предсказание на основе обучения невозможно. Рассмотрим двоичное представление $\sqrt{2}\ : e = 1.0110\ldots1_n\ldots,\ e_0 = 1, e_1 = 0, \ldots, e_n = 1$ и последовательность $\{e_i\}$. Известно, что элементы этой последовательности независимы в совокупности, поэтому невозможно предсказать значение в любом разряде $(e_n)$, на основе значений элементов последовательности в старших разрядах $(e_0, \ldots, e_{n-1})$. Иначе говоря, методы построения предиктора, рассмотренные выше, не позволят решить задачу предсказания. Тем не менее, если модель известна, то значение $e_n$ определяется точно, причём без использования $e_0, \ldots, e_{n-1}$.

В общем случае, зная точную модель, задача отыскания опти-

мального предиктора, сводится к следующей:

$$X = \arg\min_{X} Err(X), \tag{6}$$

где $Err(X)$ — ошибка прогноза. Однако если модель известна, то приближённо. Пусть имеется параметрический класс моделей и класс предикторов. Выберем произвольно модель и предиктор из класса и вычислим ошибку прогноза. Из всех моделей выберем наихудшую, т.е ту, для которой ошибка максимальна:

$$Err(X, \theta) \sim \max_{\theta}. \tag{7}$$

Решением задачи прогнозирования, будет минимаксный предиктор $X$, являющийся решением следующей задачи на минимум:

$$\max\{Err(X, \theta)|\theta\} \sim \min_{X}. \tag{8}$$

**Литература**

1. Бокс Дж., Дженкинс Г. Анализ временных рядов прогноз и управление. 1974 г.

2. Цыплаков Александр — Введение в прогнозирование в классических моделях временных рядов. // Квантиль,№1 — С. 3–19.

3. Лукашин Ю. П. Адаптивные методы краткосрочного прогнозирования временных рядов.– М.: Финансы и статистика, 2003.

4. Константин Дегтярёв. Прогнозирование валютных котировок с использованием модифицированного стационарного метода, основанного на нечетких временных рядах.

# ЦЛП-ФОРМУЛИРОВКА ДЛЯ СОВМЕЩЕННОЙ ЗАДАЧИ ВЫБОРА И ПЛАНИРОВАНИЯ ИНСТРУКЦИЙ В ГЕНЕРАТОРЕ КОДА

**Вьюкова Н.И., Галатенко В.А., Самборский С.В.**
**(Научно-исследовательский институт системных исследований РАН, Москва)**
*niva@niisi.msk.ru, galat@niisi.msk.ru, sambor@niisi.msk.ru*

## Введение

Фаза генерации кода в современных оптимизирующих компиляторах включает последовательное выполнение выбора инструкций, планирования инструкций и распределения регистров. Поскольку эти задачи взаимосвязаны, результат их последовательного решения может быть неоптимальным.

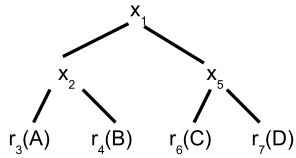
В настоящем докладе рассматривается подход, обеспечивающий одновременное выполнение выбора и планирования инструкций с учетом ограничений на число доступных регистров. Он позволяет также в случае дефицита регистров автоматически осуществлять их спиллинг (временное сохранение значения из регистра в память с последующим восстановлением). Предлагаемый метод генерации кода предполагает совместное описание задач выбора и планирования инструкций в виде задачи целочисленного линейного программирования. Он может быть использован как для линейных участков, так и для программной конвейеризации циклов.

## Метод отложенного выбора инструкций

Рассмотрим основные моменты предлагаемого метода, который получил название *генерация кода с отложенным выбором инструкций* [1]. Входными данными для генератора кода служат грамматика, описывающая систему команд целевого процессора, и дерево (или лес деревьев), задающее входной линейный участок. Далее мы будем использовать следующий простой пример грамматики:

```
N = {R, M}            /* нетерминальные символы */
A = {x, r}            /* терминальные символы */
P = {                 /* Правила */
    1 R = r :2        /* Загрузка значения на регистр */
    2 R = M :2        /* Чтение из памяти в регистр */
    3 M = R :2        /* Запись регистра в память */
    4 R = x(R,R) :3   /* Выполнение операции x */
}
```

Здесь нетерминальный символ **M** соответствует памяти, **R** – регистр. Терминальный символ **x** соответствует некоторой арифметической инструкции процессора, а **r** описывает значение в памяти или константу. Целые константы, заданные через двоеточия после правил, описывают латентность соответствующих машинных команд. Пример входного дерева для выражения $(A * B) * (C * D)$:




Индексы задают нумерацию узлов дерева. Каждому узлу сопоставляется множество виртуальных регистров: V1, M1 для узла 1, V2, M2 для узла 2, и т.д., исходя из того, что результат подвыражения в узле $i$ может быть вычислен либо в регистре (R$i$), либо в ячейке памяти (M$i$).

Генерация кода состоит из двух шагов: 1) первичный выбор инструкций и 2) планирование с окончательным выбором инструкций.

Мы используем модификацию известного метода BURG [2], одного из декларативных методов выбора команд, обзор которых можно найти в [3]. Алгоритм BURG основан на методах синтаксического разбора, где командам процессора соответствуют правила вывода в некоторой контекстно-свободной грамматике (без однозначности разбора). Применение правила имеет определенную цену; цена дерева разбора – сумма цен примененных правил, что позволяет находить оптимальный разбор методом динамического программирования.

Модификация алгоритма BURG заключается в том, что запоминаются все возможные команды, применимые в узлах дерева (а не только наиболее дешевые). Для нашего примера будут сгенерировано множество команд ($Com$):

```
Узел   Команды
-----------------------------------------
 1     R1 = x(R2,R5)     M1 = R1   R1 = M1
 2     R2 = x(R3,R4)     M2 = R2   R2 = M2
 3     R3 = r3           M3 = R3   R3 = M3
 4     R4 = r4           M4 = R4   R4 = M4
 5     R5 = x(R6,R7)     M5 = R5   R5 = M5
 6     R6 = r6           M6 = R6   R6 = M6
 7     R7 = r7           M7 = R7   R7 = M7
```

Команды вида M$i$=R$i$ и R$i$=M$i$ обычно являются избыточными. Но в условиях дефицита регистров планировщик может включить их в расписание для реализации спиллинга.

Окончательный выбор инструкций выполняется во время планирования. Формулировка ЦЛП-задачи планирования с выбором команд в целом аналогична [4]; отличия касаются методов подсчета числа требуемых регистров и требования об однократности выполнения каждой входной команды. Здесь оно заменяется требованием о необходимости выполнения подмножества команд из $Com$, достаточного для вычисления входного выражения.

В формулировке ЦЛП-задачи используются константы $T$ и $Nreg$. $T > 0$ – задает число тактов, за которое должна выполниться программа. На стадии планирования последовательно делаются попытки составить расписание, укладывающееся в $T$=$Tmin$, $Tmin + 1$, ... тактов, пока решение ЦЛП-задачи не закончится успешно. Константа $Nreg$ указывает число доступных физических регистров.

Из-за ограничений на размер тезисов мы не приводим полную формулировку ЦЛП-задачи. Рассмотрим результат ее решения в условиях острого дефицита физических регистров в предположении, что процессорная архитектура допускает запуск двух команд за такт без дополнительных условий их совместимости.

Не выталкивая регистры в память, нельзя вычислить выражение $(A * B) * (C * D)$ менее, чем на 3 регистрах (не меняя заданный скобками порядок операций). Попробуем решить построенную ЦЛП задачу для $Nreg = 2$. При $T < 13$ решения нет, но выделив 13 тактов удается сгенерировать код:

```
          S                 Вирт. регистры    Ассемблерный код
                            хранимые "недо-
                                     писанные"
 1:  R6 = r6  R7 = r7                        load  f0,C;    load  f1,D
 2:                                 R6 R7    nop
 3:  R3 = r3  R5 = x(R6,R7)  R6 R7           load  f0,A;    mul  f1,f0,f1
 4:                                 R3 R5    nop
 5:                          R3     R5       nop
 6:  R4 = r4  M5 = R5        R3 R5           load  f1,B;    store  f1,tmp_R5
 7:                          R3     R4 M5    nop
 8:  R2 = x(R3,R4)  R5 = M5  R3 R4 M5        mul  f0,f0,f1; load  f1,tmp_R5
 9:                                 R2 R5    nop
10:                          R5     R2       nop
11:  R1 = x(R2,R5)           R2 R5           mul  f0,f0,f1
12:                                 R1       nop
13:                                 R1       nop
```

Легко видеть, что сгенерированный код действительно использует всего два физических регистра. Это достигается за счет того, что на шестом такте содержимое регистра R5 сохраняется в память (и замещается значением переменной B), а на такте 8 загружается обратно. Таким образом, полностью автоматически выбран регистр для спиллинга и сгенерирован спилл-код.

**Заключение**

При одновременном выборе инструкций и их планировании несложно учесть ограничения по числу регистров и реализовать спиллинг регистров, сохранение значений на регистрах другого типа (менее дефицитных), или повторное вычисление некоторого значения, если это дешевле, чем хранение. Также автоматически реализуется дублирование значений, если это необходимо для параллельной обработки.

Обычно эти преобразования реализуются в компиляторах по отдельности, и редко полностью. Часто реализуется единственное преобразование без которого нельзя обойтись (спиллинг). При этом нет гарантии, что будет оптимально выбран регистр для выталкивания в память. В предлагаемом подходе все управление регистрами происходит комплексно и гарантировано оптимальным образом.

Для этого достаточно в формулировке совмещенной задачи выбора и планирования инструкций разрешить повторное вычисление любого виртуального регистра, а также исполнение любой инструкции более одного раза.

**Литература**

1. Вьюкова Н.И. Генерация кода с отложенным выбором инструкций – М.: НИИСИ РАН, 2010, с. 80-96.

2. Fraser C.W., Henry R.R., Proebsting T.A. BURG – Fast optimal instruction selection and tree parsing // SIGPLAN Notices 27, 4 (Apr. 1992), p. 68-76.

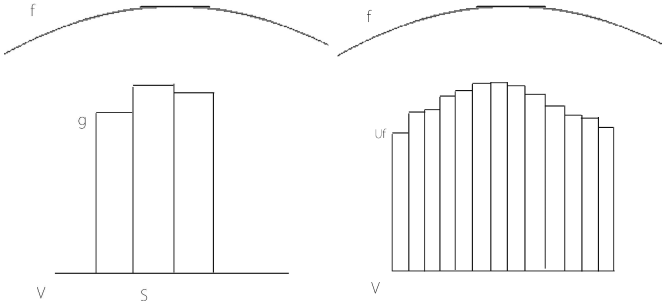
3. Вьюкова Н.И., Галатенко В.А., Самборский С.В. К проблеме выбора машинных инструкций в генераторах кода. – М.: НИИСИ РАН, 2009, с. 3-31.

4. Самборский С.В. Формулировка задачи планирования линейных и циклических участков кода // Программные продукты и системы. 3(79), 2007, с. 12-16.

## СВЕРХРАЗРЕШЕНИЕ И РОБАСТНОСТЬ ДИНАМИЧЕСКИХ МАТРИЦ СЕНСОРОВ

**Григорьева А. М., Пытьев Ю. П.**
**(МГУ имени М.В. Ломоносова)**
*elf1ka@mail.ru, yuri.pytyev@gmail.com*

Пусть на рисунке 1а) $V$ — одномерное поле зрения, $S$- строка сенсоров. $f$- распределение интенсивности потока, падающего на поле зрения $V$. $g$- выходной сигнал матрицы сенсоров, причем выходной сигнал каждого сенсора прямопропорционален потоку интенсивности, падающего на сенсор, и не содержит информации о вариациях потока в его пределах.




а) б)

Рис. 1: $V$ —поле зрения, $f$ —распределение интесивности потока, падающего на $V$, $s$ —строка сенсоров.а)$g$ —выходной сигнал строки сенсоров, б) $s$ —выходной сигнал строки сенсоров, покрывающей всё поле зрения, и состоящей из сенсоров, размеры которых равны шагу перемещения строки сенсоров.

Определим геометрическую разрешающую способность строки сенсоров как размер светочувствительного элемента. Чем меньше размер сенсора, тем выше геометрическая разрешающая способность строки сенсоров. При регистрации сигнала $g$ будем перемещать сроку сенсоров по полю зрения с шагом, меньшим чем размер сенсора, и фиксировать сигнал. Зарегистрированный сигнал методами математической редукции можно свести к виду, свойственному выходному

сигналу $Uf$(рис. 1б)) строки сенсоров, покрывающей всё поле зрения $V$, и состоящей из сенсоров, размеры которых равны шагу перемещения строки сенсоров. Таким образом решается задача сверхразрешения — восстанавливается вариация интенсивности потока $f$ до размера, фиксированного шагом перемещения строки сенсоров по полю зрения. Такая система регистрации обладает так же робастностью. Представим, что один из сенсоров строки вышел из строя. Благодаря тому, что строка сенсоров перемещается, интенсивность потока света, падающего на область неработающего сенсора будет зарегистрирована другими, работающами сенсорами. При этом, по виду сигнала $Uf$ нельзя определить какой сенсор вышел из строя, и система регистрации будет продолжать работать.

Для компьютерного моделирования данной системы удобнее регистрировать сигнал не строкой сенсоров, а одним сенсором. Тогда строка сенсоров будет смоделирована таким образом: зафиксируем положения и время нахождения сенсоров строки, затем будем последовательно регистрировать сигнал, перемещая сенсор по полю зрения и задерживаясь в каждой позиции на соответствующее время. Система с одним сенсором универсальна с той точки зрения, что мы можем регулировать время нахождения сенсора в заданной позиции поля в зависимости от того, на сколько хотим понизить уровень шума в данной области поля зрения $V$. Но для регистрации динамических потоков света лучше объединять сенсоры в матрицу, так как регистрация сигнала одним сенсором требует больше времени.

Данная система регистрации описывается следующей схемой измерений:

$$\begin{pmatrix} g_1 \\ \vdots \\ g_p \end{pmatrix} = \begin{pmatrix} A_1 \\ \vdots \\ A_p \end{pmatrix} f + \begin{pmatrix} \nu_1 \\ \vdots \\ \nu_p \end{pmatrix}, \tag{1}$$

где $g$ и $\nu$ — независимые случайные векторы с математическим ожиданием равным нулю и ковариационными операторами $P$ и $K$ соответственно. Вектор $f$ моделирует распределение интенсивности светового потока, падающего на поле зрения. $A_i$ — оператор, моделирующий отклик матрицы сенсоров на световой поток $f$ в каждом акте регистрации, $i = 1, \ldots p$. Вектор $\nu$ моделирует аддитивный шум.$g_i$ — выходной сигнал матрицы сенсоров. Нужно оценить выходной сигнал $Uf$ матрицы, состоящей из сенсоров с размерами равными шагу

сканирования. Для этого минимизируем функционал:

$$\|Rg - Uf\|^2 \sim \min_R, \tag{2}$$

где $R$ — оператор редукции.

Для двумерного поля зрения задача большой, поэтому для вычисления оператора $R$ восспользуемся рекуррентной редукцией:

$$Uf_i = Uf_{i-1} + \frac{UP_iA_i\,(g_i - A_if)}{{\sigma_i}^2 + A_i^*P_iA_i}, \tag{3}$$

$$P_i = P_{i-1} + \frac{P_{i-1}A_i(P_{i-1}A_i)^*}{{\sigma_i}^2 + A_i^*P_iA_i}, \tag{4}$$

где $i = 1, \ldots p$, $\widehat{g_0'} = 0$, $P_0 = P$, $K = diag(\sigma_1^2, \ldots \sigma_p^2)$ — ковариационный оператор шума.

На рисунке 2 показано изображение, которое получено с помощью статической матрицы сенсоров с низким разрешением, покрывающей всё поле зрения.

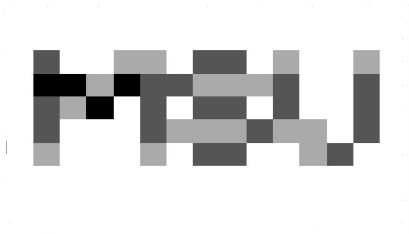


Рис. 2: Изображение, полученное статической матрицей сенсоров, покрывающей всё поле зрения

На рисунках 3а),б)представлено изображение, полученное методом математической редукции сигнала, зарегестрированного динамической матрицей сенсоров, причем рисунку 3а) соответствует большее время регистрации сигнала, чем изображению, представленному на рисунке 3б).

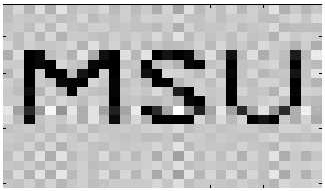
Рисунки 4а),б) демонстрируют робастность описанной выше системы регистрации изображений. Изображение на рисунке 4а)было

а)

б)

Рис. 3: Изображение, полученное методом математической редукции сигнала

получено методом математической редукции сигнала, зарегестрированного стопроцентно работающей матрицей, а на рисунке 4б) матрицей, часть сенсоров которой вышли из строя.



а)

б)

Рис. 4: Изображение, полученное методом математической редукции

Рассмотренная система регистрации подобна зрительной системе человека. Глазное яблоко при регистрации изображений совершает непрерывные неконтролируемые человеком движения такие как дрейф, скачки и тремор. И разрешающая способность зрительной системы дейтвительно выше, чем геометрическая разрешающая способность сетчатки, которая определяется размерами колбочек. Причем гибель отдельных колбочек не приводит к заметным дефектам зрения.

**Литература**

1. Ярбус А. Л. Роль движений глаз в процессе зрения — М.: Наука, 1965. — 167 с.

2. Демидов В. Е. Построение оценок. — М.: Наука, 2008.

3. Пытьев Ю. П. Методы математического моделирования измерительно-вычислительных систем. — Москва: ФИЗМАТЛИТ, 2011. — 400 с.

# ЗАДАЧА МИНИМИЗАЦИИ ЧИСЛА ПАРАЛЛЕЛЬНЫХ ПРОЦЕССОРОВ ДЛЯ СИСТЕМЫ ЗАДАНИЙ С ОГРАНИЧЕНИЯМИ ПРЕДШЕСТВОВАНИЯ

**Григорьева Н. С.**
**(С.Петербургский государственный университет)**
*gns@interzet.ru*

Рассмотрим систему заданий $U = \{u_1, u_2, \ldots, u_n\}$, на которой отношение частичного порядка $\prec$ задано графом $G = \langle U, E\rangle$, в котором есть дуга $e = (u_i, u_j) \in E$ тогда и только тогда, когда $u_i \prec u_j$. Если $u_i \prec u_j$ то выполнение задания $u_j$ может быть начато только после завершения задания $u_i$. Задано время выполнения каждого задания $t(u_i)$, которое будем считать целым.

Множество заданий выполняется на параллельных идентичных процессорах, любое задание может выполняться на любом процессоре, и каждый процессор может выполнять не более одного задания в каждый момент времени. Прерывания выполнения заданий не допускаются. Для данной модели можно рассмотреть две задачи составления расписаний [1]. В задаче 1 задано число процессоров, а время выполнения заданий требуется минимизировать. В задаче 2 общее время выполнения заданий задано, требуется минимизировать количество процессоров. Эти задачи, за исключением некоторых частных случаев, являются $NP-$трудными [2]. Для решения этих двух задач предлагается метод ветвей и границ в сочетании с бинарным поиском. Алгоритм решения задачи 1 рассмотрен в [3], ниже рассмотрим задачу 2. Общей подзадачей этих двух задач будет задача построения допустимого расписания при заданном числе процессоров и общем времени выполнения.

Построить расписание, значит найти для каждого задания $u_i$ время начала выполнения задания $\tau(u_i)$ и номер процессора $num(u_i)$, на котором оно выполняется. Длиной расписания $S$ называется величина

$$T_S = \max\{\tau(u_i) + t(u_i) | u_i \in U\}.$$

Требуется определить минимальное число процессоров $M_{opt}$ при котором существует расписание $S$ заданной длины $T_S$.

**Алгоритм нахождения минимального числа процессоров**

Пусть $M_{opt}$ минимальное количество процессоров. Определим интервал $(a, b]$ такой, что $a < M_{opt} \leq b$. Нижняя оценка значения целевой функции в данной задаче $LB(M) = \lceil \sum_{i=1}^{n} t(u_i)/T_S \rceil$. Поэтому положим $a = LB(M) - 1$. В качестве верхней оценки возьмем $b = n$. Тогда $M_{opt} \in (a, b]$.

**Алгоритм построения расписания** $SCH(U, T_S, a, b; S, M_{opt})$

Пока $b - a > 1$ положим $m := \lceil (a + b)/2 \rceil$. Решим задачу построения допустимого расписания для $m$ процессоров процедурой $BB(U, m; S)$. Если расписание $S$ для $m$ процессоров найдено, то $b := m$, в противном случае $a := m$. В результате определяется минимальное количество процессоров $M_{opt}$, при котором существует расписаний $S$ заданной длины.

**Метод ветвей и границ построения допустимого расписания** $BB(U, m; S)$

Требуется построить допустимое расписание $S$ длиной $T_S$ при заданном количестве процессоров $m$ или определить невозможность построения такого расписания.

Для каждого задания $u$ определяется раннее время начала выполнения задания $v_{min}(u)$, и позднее время начала задания $v_{max}(u)$. Возможное время простоя процессоров $I = m \cdot T_S - \sum_{i=1}^{n} t(u_i)$.

Для частичного решения $\sigma_k = (u_{i_1}, u_{i_2}, \ldots, u_{i_k})$ известно время освобождения процессоров после выполнения заданий, включенных в частичное решение, $time_k[1:m]$ и $t_{min}(k) = \min\{time_k[i] | i \in 1:m\}$. Для каждого задания $u \in \sigma_k$ определено время начала его выполнения $\tau(u)$, номер процессора $num(u)$, на котором оно выполняется, время простоя процессора перед его началом $r(u)$, резерв простоев $I_k = I - \sum_{i=1}^{k} r(u_i)$ и $R_k$ - множество заданий запрещенных на уровне перебора $k$, тех заданий, которые уже добавлялись к частичному решению и привели к недопустимым частичным решениям.

Основным способом сокращения перебора будет как можно более раннее установление недопустимости частичного решения.

**Условия недопустимости частичного решения** $\sigma_k$

1. Если существует задание $u_{cr} \notin \sigma_k$, такое что $v_{max}(u_{cr}) < t_{min}(k)$, то частичное решение $\sigma_k$ недопустимо.

2. Рассмотрим интервалы времени $[t_1, t_2] \subseteq [t_{min}(k), T_S]$. Для каждого интервала определим общую мощность процессоров,

которые свободны в данном интервале $MP(t_1, t_2)$. Для каждого задания $u_i \notin \sigma_k$, вычислим $v_i = \max\{v_{min}(u_i), t_{min}(k)\}$ возможное время начала задания и найдем интервалы $x_i = [v_i, v_i + t(u_i)] \cap [t_1, t_2]$, и $y_i = [v_{max}(u_i), v_{max}(u_i) + t(u_i)] \cap [t_1, t_2]$,. Обозначим $L([t_1, t_2])$ длину интервала $[t_1, t_2]$.Найдем $M_k(t_1, t_2)$ минимальную потребность заданий в ресурсах на интервале $[t_1, t_2]$, и $M_k(t_1, t_2) = \sum_{u_i \notin \sigma_k} min\{L(x_i), L(y_i)\}$.

Если нашелся интервал $[t_1, t_2]$, такой что $M_k(t_1, t_2) > MP(t_1, t_2)$, то частичное решение $\sigma_k$ недопустимо.

3. Частичное решение $\sigma_k$ недопустимо, если нет заданий $u \notin R_k$, для которых все предшественники $u$ включены в частичное решение $\sigma_k$, и выполнено $v_{min}(u) - t_{min}(k) \leq I_k$.

4. Если существует задание $u_{cr} \notin \sigma_k$, такое что $v_{max}(u_{cr}) < t_{min}(k)$, тогда для любого задания $u$, такого что

$$\max\{t_{min}(k-1), v_{min}(u)\} + t(u) > v_{max}(u_{cr})$$

частичное решение $\sigma_{k-1} \cup u$ недопустимо.

5. Если существует задание $u_{cr} \notin \sigma_k = \sigma_{k-1} \cup u_k$, такое что $v_{max}(u_{cr}) < t_{min}(k)$, и выполнены неравенства $v_{max}(u_k) < t_{min}(k)$ и $t_{min}(k-1) + t(u_{cr}) > v_{max}(u_k)$, тогда недопустимо частичное решение $\sigma_{k-1}$.

Доказательства утверждений приведены в [3]. Второе условие является модификацией оценок, предложенных в [4].

**Алгоритм построения допустимого расписания** $BB(U, m; S)$**.**

$k = 0$; $time[i] = 0; i \in 1 : m$ $\sigma_0 = \emptyset$;

Для частичного решения $\sigma_k$ выполнить:

1. Проверить для всех заданий $u \notin \sigma_k$, выполнение условия 1. Если оно не выполнено, то $u_{cr} := u$ и перейти к п.6.

2. Если не выполнено условие 2, то перейти к п.6.

3. Если не выполнено условие 3, то перейти к п.5.

4. Выбрать задание $u_0$, и добавить к частичному решению $\sigma_k$. $k := k + 1$. Если $k = n$, то конец алгоритма, иначе перейти к п.1.

5. Аннулировать список заданий, запрещенных на $k-$ ом уровне перебора $R_k := \emptyset$.

6. Если $k = 0$, то конец алгоритма. Шаг назад: от частичного решения $\sigma_k = \sigma_{k-1} \cup u_t$ вернуться к решению $\sigma_{k-1}$. Задание $u_t$ перевести во множество $R_k$ и $k := k - 1$.

7. Если выполнено условие 4, то запретить группу заданий, если выполнено условие 5, то сделать еще один шаг назад. Перейти к п.1.

Если $k = 0$, то допустимого расписания не существует, если $k = n$, то получили допустимое расписание $S = \sigma_n$, с длиной $T_S$.

**Приближенный алгоритм**

Для уточнения верхней оценки числа процессоров можно применить приближенный алгоритм. Определим минимально возможное количество процессоров $m$ и будем строить допустимое расписание $S$ заданной длины $T_S$, проверяя на каждом шаге выполнение условий 1 и 2 допустимости расписания. Если будет установлено, что частичное расписание $\sigma_k$ недопустимо, то увеличим число процессоров на единицу и начнем построение расписания $S$ сначала уже для большего числа процессоров.

Вычислительный эксперимент подтвердил работоспособность алгоритмов.

**Литература**

1. Теория расписаний и вычислительные машины. Под ред. Коффмана Э. Г. –М.:Наука,1984.( Computer and job-shop Scheduling theory. Ed. by E.G. Coffman, J.)

2. Ullman J. D. NP-complete scheduling problems//Journal Comput. System Sci. –1975,10, P. 384-393.

3. Григорьева Н. С. Алгоритм ветвей и границ построения расписания для многопроцессорной системы// Вестник СПбГУ, сер. 10. – 2009, вып.1, с. 44-55.

4. Fernandez E., Bussell B. Bounds the number of processors and time for multiprocessor optimal schedules//IEEE Trans. on Computers. –1973, P. 745-751.

# ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ МЕТОДА АКТИВНЫХ КОНТУРОВ

**Жизневский А.Н. (Казанский (Приволжский) Федеральный Университет)**
*write2aracon@gmail.com*

**Введение**

При дешифрировании изображений часто используются геометрические признаки. При этом обязательным этапом дешифрирования является получение проекций объектов (сегментация).

В 1988 г. в работе [1] предложен новый подход к задаче поиска границ, названный методом активных контуров (или методом «змей»). В соответствии с ним на плоскости задается семейство кривых, содержащее границу заданного объекта. На этом семействе задается целевая функция, принимающая минимальное значение на границе. Граница выявляется методом последовательных приближений.

Вслед за этой публикацией разными авторами был предложен ряд модификаций и новых методов на основе идеи активных контуров [2-7].

К сожалению, практическое применение этого метода наталкивается на ряд серьезных трудностей, обсуждению которых посвящена настоящая работа.

**1. Описание метода** Пусть $v(s) = (x(s), y(s))$, $s \in [0, 1]$ — кривая, параметризованная по длине. Тогда целевая функция (энергия) определяется следующим образом:

$$E(v) = \int_0^1 \left( \frac{1}{2}\alpha(s)||v'_s(s)||^2 + \frac{1}{2}\beta(s)||v''_{ss}(s)||^2 + E_{ext}(v) \right) ds,$$

Первое слагаемое в скобках растёт при увеличении длины кривой, второе — при увеличении кривизны. Коэффициенты $\alpha(s)$ и $\beta(s)$ определяют вклад длины и кривизны в общую энергию кривой.

В [1] предлагаются различные виды $E_{ext}(v)$. Для поиска границ предлагается использовать градиент сглаженного изображения:

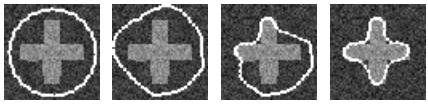
$$E_{ext}(v) = -||\nabla[G_\sigma(x(s), y(s)) * I(x(s), y(s))]||^2$$

Необходимым условием экстремума $E$ является выполнение урав-

нения Эйлера [8]:

$$-(\alpha(s)v'_s(s))'_s + (\beta(s)v''_{ss}(s))''_{ss} + \nabla E_{ext}(v) = 0$$

Это уравнение решается численно методом последовательных приближений. Никаких рекомендаций по выбору начального приближения и критерию остановки вычислений не предлагается.



а) б) в) г)

Рис. 1: Работа метода активных контуров. $\alpha = 0,03$, $\beta = 0,0001$, $\sigma = 2,0$.

На рис. 1 представлен пример движения активного контура по изображению размера $60 \times 60$ пикселов. Объект (крест) имеет размер $40 \times 40$ пикселов и расположен в центре изображения. Здесь и далее значения энергетической яркости точек объекта и фона имеют нормальное распределение со стандартным отклонением 20 и средним значением 60 (фон) и 140 (объект). В качестве начального приближения для метода выбрана окружность (рис. 1, а). На рис. 1 (б,в,г) показаны положения контура после 1000, 3000 и 5000 итераций соответственно. На последующих итерациях значительного изменения в положении контура не наблюдалось.

**2. Результаты экспериментов**

Важным достоинством подхода является учет замкнутости границы и ограничение кривизны. Кроме того, найденная проекция автоматически получается односвязной. В настоящее время часто используется задание границы с помощью множества уровня (level-set), позволяющее искать объекты с многосвязными проекциями (см., например, [9]).

При моделировании выявлены два серьезных недостатка метода. Первый связан с зависимостью результата сегментации от выбора начального приближения. На рис. 2 показаны результаты сегментации изображения при двух вариантах выбора начального приближения. В качестве начальных приближений были выбраны окружности, расположенные в центре изображения, с диаметром 54 пиксела

а) б) в) г) д)

Рис. 2: Зависимость результата сегментации от выбора начального приближения. $\alpha = 0.005$, $\beta = 0.00005$, $\sigma = 1.0$.

(б) и 48 пикселов (г). Результаты сегментации представлены соответственно на рис. 2, (в, д). Как можно видеть, они существенно отличаются. Методы из более поздних работ, например, GVF [3], позволяют уменьшить зависимость результата от начального приближения.

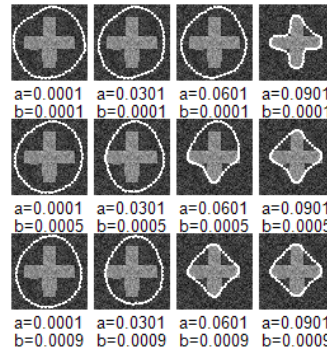



Рис. 3: Зависимость результата сегментации от выбора значений коэффициентов $\alpha$ и $\beta$. $\sigma = 1,0$.

Второй недостаток метода заключается в отсутствии объективного способа выбора коэффициентов $\alpha$ и $\beta$, которые могут сильно повлиять на результаты сегментации. На рис. 3 показаны положения активного контура после 30000 итераций при различных значениях $\alpha$ и $\beta$. Во всех экспериментах использовалось одно и то же изображение и одно и то же начальное приближение в виде окружности с диаметром 54 пиксела, расположенной в центре изображения. Результаты сегментации для разных $\alpha$ и $\beta$ существенно отличаются.

Описанные недостатки проявляются и в случае объектов более простых форм (например, квадрат).

**Заключение**

Проведено экспериментальное исследование метода активных контуров применительно к сегментации изображений. Выявлены зависимость результата сегментации от выбора начального приближения и коэффициентов $\alpha$ и $\beta$.

**Литература**

1. Kass M., Witkin A., Terzopoulos D. Snakes: Active contour models // Int. J. Comput. Vis., 1988. № 1. С. 321–331.

2. Cohen L. D. On active contours models and balloons // Computer Vision, Graphics, and Image Processing: Image Understanding. 1991. 53(2). С. 211–218.

3. Xu C., Prince J.L. Gradient vector flow: A new external force for snakes. // IEEE Proc. Conf. on Comp. Vis. Patt. Recog. (CVPR'97), 1997, с. 66–71.

4. Williams D. J., Shah M. A fast algorithm for active contours and curvature estimation. // CVGIP: Image Underst., 55(1), 1992, с. 14—26.

5. Amini A. A., Weymouth T. E., Jain R. C. Using dynamic programming for solving variational problems in vision. // IEEE Transactions on pattern analysis and machine intelligence, vol.12, №9. 1990, С. 855–867.

6. Melonakos J., Pichon E., Angenent S., Tannenbaum A. Finsler Active Contours. // IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, №3, 2008, С. 412–423.

7. Caselles V., Kimmel R., Sapiro G. Geodesic active contours. // International journal of computer vision, vol. 22, №1, 1997, С. 61–79.

8. Эльсгольц Л. Э. Дифференциальные уравнения и вариационное исчисление. — М.: Наука, 1965.

9. Chan T. F., Vese L. A. Active Contours Without Edges. // IEEE Trans. on Image Processing. № 2, 2001, С. 266–277.

# ЛОКАЛЬНЫЙ ЭЛИМИНАЦИОННЫЙ АЛГОРИТМ И ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

**Лемтюжникова Д.В., Щербина О.А.**
**(Таврический национальный университет имени В.И. Вернадского)**
*darabbt@gmail.com, oshcherbina@gmail.com*

Из-за высокой вычислительной трудоемкости, а также возможности распараллеливания вычислительных процессов, присущей алгоритмам дискретной оптимизации (ДО), разработка параллельных алгоритмов для решения задач ДО представляет значительный интерес. Параллельные вычислительные системы обладают возможностями параллельного использования большого числа процессоров для обработки информации. Разумное использование вычислительного потенциала посредством распараллеливания вычислительного процесса существенно ускоряет решение практически важных задач ДО большой размерности. Современные достижения параллельных методов комбинаторной оптимизации описаны в [1]. Декомпозиционные алгоритмы ДО являются первоочередными кандидатами для параллелизации [2, с. 225, 257].

Для решения задач ДО с помощью декомпозиционных методов представляют интерес *системы с распределенной памятью*. Параллельное программное обеспечение (ПО) для этих систем использует явную декомпозицию решаемой задачи на подзадачи и назначение их процессорам, а также быструю коммуникационную сеть, обеспечивающую синхронный обмен данными между процессорами. Наиболее типичной архитектурой является Беовульф кластер, важные свойства которого: однородность, коммуникационная сеть типа «все-со-всеми», отсутствие общей памяти и отсутствие глобальных часов для синхронизации вычислений. В такой архитектуре все процессы передачи сообщений и синхронизации должны осуществляться с помощью явного обмена сообщениями между процессорами, используя "message-passing" протокол. Программирование для системы с распределенной памятью осуществляется с помощью стандартного языка высокого уровня типа C++ или Fortran с явной передачей сообщений. Имеются два стандартных API для передачи сообщений: Message-Passing Interface (MPI) и Parallel Virtual Machine (PVM).

Остановимся на вопросах разработки ПО для параллельных алгоритмов решения задач ДО. SYMPHONY [3] – это параллельная си-

стема ПО для решения задач смешанного целочисленного линейного программирования (СЦЛП), подобная COIN/BCP [4]. COIN/BCP и SYMPHONY объединены в новом решателе ALPS [5]. Кроме того, существуют такие параллельные решатели, как PUBB [6], PPBBLib [7] и PICO [8]. PARINO [9] и FATCOP [10] – параллельные решатели общего назначения для решения задач СЦЛП, причем FATCOP предназначен для грид-систем. ParaLEX [11] – это параллельное расширение решателя CPLEX для задач СЦЛП, предназначенное для работы в распределенной вычислительной среде. Более современной системой является программная система CHiPPS (COIN-OR High Performance Parallel Search), созданная для параллелизации поисковых алгоритмов на дереве [12].

При параллельной реализации локального элиминационного алгоритма (ЛЭА) [13] предпочтительны методы top-down, работающие с макрографами потоков данных, так как структура графа потоков данных известна (элиминационное дерево или древовидная декомпозиция). Высокая вычислительная сложность, один из основных недостатков ЛЭА, может быть снижена с помощью параллельных вычислений. ЛЭА основан на использовании элиминационного дерева, которое обладает необходимой информацией о зависимости между данными в задаче ДО. Высота элиминационного дерева представляет собой (грубую) меру вычислительной работы по параллельной элиминации. Элиминационное дерево управляет процессом элиминации переменных и блоков переменных и указывает потоки данных, а также характеризует возможности распараллеливания при решении задачи ДО с помощью ЛЭА. В случае вычислительных машин с распределенной памятью после построения элиминационного дерева далее необходимо решить задачу отображения переменных на процессоры. Основными целями этого отображения являются хорошая балансировка нагрузки и низкая межпроцессорная коммуникация, причём они могут конфликтовать друг с другом. При параллельной реализации блочного ЛЭА используемый граф потоков данных является деревом, которое должно обрабатываться от листьев к корню. Когда для вершины дерева найдено частичное решение, множество промежуточных данных передается в родительскую вершину. Как только промежуточная информация от всех вершин-потомков собрана в родительской вершине, она используется при решении соответствующей родительской вершине задачи ДО. Таким образом,

независимые ветви дерева могут обрабатываться параллельно; будем называть этот тип параллельности (древовидной) параллельностью первого типа. В общем случае, он может быть использован более эффективно в нижней части дерева, нежели вблизи ее корневой вершины. Второй тип параллельности соответствует параллельным задачам для одного блока. При этом используется парадигма "Master-Worker", в которой имеются два различных типа процессоров: мастер-процессор и рабочие процессоры. При этом управление алгоритмом осуществляется мастер-процессором: он разбивает задачу на отдельные подзадачи и назначает их для решения рабочим процессорам, учитывая взаимосвязи между ними. Такая парадигма особенно полезна, когда задача разбита на легко решаемые подзадачи, а также при слабой взаимосвязи между этими подзадачами. Мастер-процессор выбирается в начале решения пакета задач, соответствующих блоку древовидной декомпозиции. Далее, рабочие процессоры выбираются мастером динамически из списка процессоров-кандидатов, используя критерий хорошей балансировки нагрузки.

Грид-компьютинг (или метакомпьютинг) в общем описывает параллельные вычисления на географически распределенной и гетерогенной платформе [14]. При этом могут использоваться рабочие станции с общей памятью, узлы кластеров ПК, кластеры и суперкомпьютеры. Потенциал грид-компьютинга [14] используется в настоящее время в ДО лишь частично. Декомпозиционные алгоритмы решения задач дискретной оптимизации (включая ЛЭА) хорошо подходят для грид-платформ.

При разработке параллельных версий ЛЭА представляется перспективным использование грид-компьютинга с использованием интерфейса Condor, а также технология PVM.

**Литература**

1. Ferreira A., Pardalos P., eds. Solving Combinatorial Optimization Problems in Parallel: Methods and Techniques //LNCS 1054, State–of–the–Art Surveys, Springer–Verlag, 1996.

2. Bertsekas D. P., Tsitsiklis J. N. Parallel and Distributed Computation: Numerical Methods. – Englewood Cliffs: Prentice–Hall, 1989.

3. Ralphs T. K., Ladanyi L., Salzman M. J. Parallel branch, cut and price for large-scale discrete optimization // Mathematical Programming. – 2003. – B. 98. – P. 253-280.

4. Ralphs T. K., Ladanyi L. COIN/BCP User's Guide. – 2001. URL: http://www.coin-or.org (дата обращения 30.07.2011).

5. Ralphs T. K., Ladanyi L., Saltzman M. J. A library hierarchy for implementing scalable parallel search algorithms // Journal of Supercomputing. – 2004. – V. 28(2). – P. 215-234.

6. Shinano Y., Higaki M., Hirabayashi R. Control schemas in a generalized utility for parallel branch and bound // Proc. of the 1997 Eleventh International Parallel Processing Symposium. – Los Alamitos: IEEE, Computer Society Press, 1997.

7. Tschoke S., Polzer T. Portable parallel branch–and–bound library PPBBLib user manual. – Department of computer science, Univ. of Paderborn, 1996.

8. Eckstein J., Phillips C. A., Hart W. E. PICO: An Object–Oriented Framework for Parallel Branch and Bound. – RUTCOR Research Report 40–2000, Rutgers University, Piscataway, 2000.

9. Linderoth J. T. Topics in Parallel Integer Optimization. – Ph.D. Dissertation. Georgia Institute of Technology School of Industrial and Systems Engineering, 1998.

10. Chen Q., Ferris M. C., Linderoth J. T. Fatcop 2.0: Advanced features in an opportunistic mixed integer programming solver // Annals of Operations Research. – 2001. – V. 103. – P. 17-32.

11. Shinano Y., Fujie T. ParaLEX: A parallel extension for the CPLEX mixed integer optimizer // Recent Advances in Parallel Virtual Machine and Message Passing Interface // F. Cappello, T. Herault, J. Dongarra, eds.
– Springer, 2007. – P. 97-106.

12. Xu Y., Ralphs T. K., Ladanyi L., Saltzman M. J. Computational experience with a software framework for parallel integer programming // The INFORMS Journal on Computing. – 2009. – V. 21. – P. 383-397.

13. Щербина О. А. Локальные элиминационные алгоритмы решения разреженных дискретных задач // Журнал вычислительной математики и математической физики. – 2008. – Т. 48, N 1. – С. 161-177.

14. Foster I., Kesselman C. (eds). The Grid: Blueprint for a New Computing Infrastructure. San Francisco: Morgan Kaufmann Publishers Inc., 1998. 677 p.

# О ЧАСТОТНО ВРЕМЕННЫХ ПРИЗНАКАХ МНОГОКАНАЛЬНЫХ ЭЛЕКТРОЭНЦЕФАЛОГРАММ МОЗГА ПРИ ЗАБОЛЕВАНИИ ПАРКИНСОНА НА РАННЕЙ СТАДИИ

**Обухов Ю.В., Королев М.С. (ИРЭ им. В.А. Котельникова РАН)**
*obukhov@cplire.ru, korolevclub@mail.ru*

**Введение**

Важной проблемой является диагностика болезни Паркинсона (БП) на ранней стадии. Одним из методов диагностики является электроэнцефалография (ЭЭГ). Спектры сигналов ЭЭГ и магнитной энцефалографии пациентов с диагнозом БП было принято анализировать с помощью Фурье преобразования [1, 2]. Но Фурье анализ ЭЭГ не позволяет исследовать динамику электрической активности мозга. Следует отметить, что характерной чертой паркинсонизма обычно признается синдром дезинтеграции, проявляющийся на разных системных уровнях (двигательные нарушения, вегетативная, нейрогуморальная дезинтеграция, эмоциональные и психические нарушения) [3, 4]. Полагают, что при ранних формах БП эти нарушения носят преимущественно функциональный нейродинамический характер. Предполагается, что такая дезинтеграция может отражаться также в динамике электрической активности мозга. В данной работе описан метод анализа вейвлет спектрограмм ЭЭГ, направленный на поиск частотно-временных признаков раннего паркинсонизма. Суть метода заключается в том, что исследуется распределение локальных максимумов вспышек вейвлет преобразования по частоте и времени.

**Метод**

В работе исследовалась фоновая электрическая активность мозга, т.е. испытуемый сидел в кресле с закрытыми глазами в расслабленном состоянии. Было обследовано шестнадцать людей с БП 1-ой стадии по шкале Хен-Яра (ХЯ) [5], четырнадцать людей со 2-ой стадией, восемь людей $2.5-3$ стадией, и одиннадцать здоровых испытуемых. ЭЭГ записывалась с электродов со стандартным расположением 10x20 с референтными электродами на ушах. Оцифрованные записи ЭЭГ были обработаны фильтром Баттерворта 4-ого порядка для удаления частот 50, 75 и 100 Гц, а также частот ниже 1 Гц. Исследование частотно – временной динамики электрической активности проводилось с помощью вейвлет преобразования ЭЭГ:

$$W(\tau,T)=\frac{1}{\sqrt{T}}\int x(t)\psi\left(\frac{t-\tau}{T}\right)dt$$

В качестве материнской функции используется комплексный вейвлет Морле:

$$\psi(\eta)=\frac{1}{\sqrt{\pi F_b}}e^{2i\pi F_c\eta}e^{-\frac{\eta^2}{F_b}}$$

где, Fb = Fc = 1.

Вейвлет спектрограмма (Рис. 1) состоит из временных серий вспышек, расположенных в различных частотных диапазонах – тета, альфа, бета и др. Суть метода заключается в том, что исследуется распределение локальных максимумов вспышек вейвлет преобразования по частоте и времени. В качестве примера на Рис. 1 представлено вейвлет преобразование сигнала в центральном отведении C3 здорового испытуемого 27 лет. Для нахождения местоположения вспышек на плоскости частота - время делается две проекции вейвлет преобразования – одна на плоскость частота – амплитуда, другая – на плоскость время - амплитуда. С помощью первой проекции выделяются частотные диапазоны хребтов вейвлет спектрограмм. Далее, в каждом из диапазонов частот вейвлет спектрограмма проецируется на плоскость время – амплитуда. Это необходимо для того, чтобы вспышки высоких амплитуд одного хребта не перекрывали вспышки других диапазонов. Далее вычисляются координаты положения всех вспышек. Таким образом, задача поиска локальных максимумов вейвлет - преобразования на плоскости частота – время сводится к поиску локальных максимумов в двух плоскостях – время – амплитуда и частота - амплитуда.

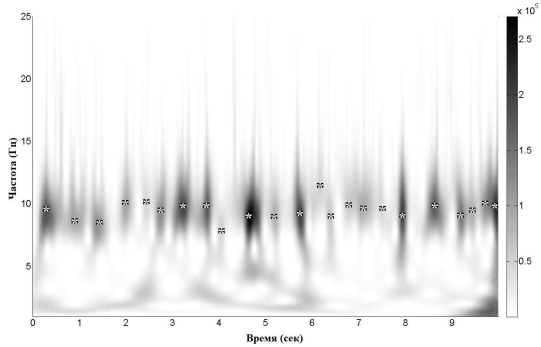
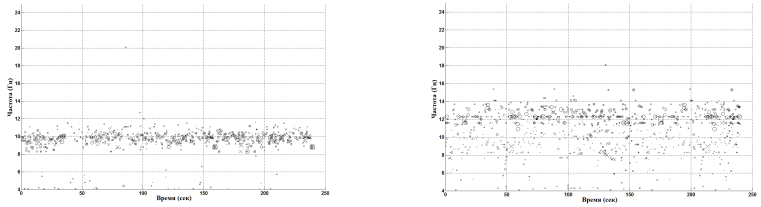


Рис.1. Вейвлет спектрограмма ЭЭГ здорового испытуемого с отведения C3 в полосе частот 1-25 Гц.

В результате, получены 2 координаты локальных максимумов и значения их амплитуд. На рис. 2 представлены положения локальных максимумов на плоскости время-частота с симметричных пар отведений C3 (в виде крестиков) и C4 (в виде ноликов), отвечающих за моторику, здорового испытуемого и пациента с заболеванием Паркинсона на 1-ой стадии.



а) б)

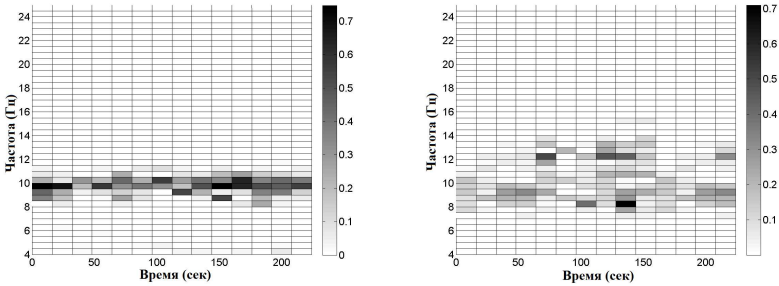
Рис. 2. а). Испытуемый К. (здоровый человек, 27 лет), б) Испытуемый Е. (пациент на 1-ой стадии БП, 29 лет) X - отведение C3, O - отведение C4

Для анализа динамики изменения частоты доминирующего ритма во времени в каждой полосе частот 0,5 Гц суммируются амплитуды локальных максимумов вспышек в окне 16 секунд и сдвигом окна на 16 секунд на всем временном интервале продолжительностью 240 секунд. Соответствующие гистограммы распределений частот вспышек с симметричных пар отведений ЭЭГ отображаются на одном графике (Рис. 3).

Из рис. 3 видно, что у здорового человека вспышки распределены в узком диапазоне частот от 9 до 11 Гц. На ранней стадии паркинсонизма происходит увеличение частоты основного ритма в левом полушарии (X), уширение диапазона частот и проявляется межполушарная асимметрия электрической активности.

**Заключение**

Разработан новый метод частотно - временного анализа электрической активности мозга, ориентированный на поиск признаков заболевания Паркинсона на ранней стадии. Впервые обнаружено повышение частоты и доминирующего ритма альфа диапазона на ранней




а) б)

стадии паркинсонизма. Метод позволяет обнаружить межполушарную асимметрию и разброс по частоте доминирующего ритма, что соответствует односторонней форме на ранней стадии заболевания и представлениям о дезорганизации электрической активности мозга при паркинсонизме.

Работа поддержана Программой Президиума РАН «Фундаментальные науки - медицине»

**Литература**

1. Stoffers D., Bosboom J. L. W., Dejen J. B., Wolters E. C., Berendse H. W., Stam C. J. Slowing of oscillatory brain activity is a stable characteristic of Parkinson's disease without dementia. Brain, 2007; 130: 1847-1860

2. Soikkeli, R., Partenen J., Soininen H., Paakkonen A., Riekkenen P. Sr. Slowing of EEG in Parkinson's disease. EEG Clin. Neurophysiol., 1991; 79:159-165

3. Вейн А. М., Голубев В. Л., Яхно Н. Н. Паркинсонизм с позиций функционально-неврологического анализа // Паркинсонизм: Вопросы клиники, патогенеза и лечения. М., 1974. С. 57–65.

4. Голубев В. Л., Левин Я. И., Вейн А. М. Болезнь Паркинсона и синдром паркинсонизма. М.: МЕДпресс, 1999. 415 с.

5. Movement Disorder Society Task Force on Rating cales for PD. Unified Parkinson's disease rating scale: status and recommendations. Mov. Disord. 2003:18:738 –750

# МОДЕЛЬ ИНФОРМАЦИОННО-ПОИСКОВОГО САЙТА ФИЛИАЛА МГУ

**Осокин В.В. (Московский Государственный Университет им.М.В.Ломоносова)**
*osvic@mail.ru*

Ставится задача построения универсальной информационно-поисковой системы, на базе которой можно реализовать веб-сайт любого филиала МГУ, факультета или кафедры. Система должна включать в себя объекты разных типов, таких как, например, сотрудник, студент, курс, практикум и т.д. В дальнейшем будем называть такие объекты *сущностями*. Система должна обеспечивать авторизированным пользователям интерфейс добавления произвольных *типов сущностей* с произвольными полями. Также система должна обеспечивать авторизированным пользователям интерфейс добавления произвольных сущностей введенных в систему типов.

Базовым типом сущности системы является *веб-страница*. Сущности такого типа имеют всего два поля: название и html-описание сущности. Все остальные типы сущности «наследуют» данный тип: в дополнение к указанным двум полям для каждого типа может быть указано неограниченное число дополнительных полей. Так, для сущностей типа «человек» логично добавить поле «дата рождения».

Сущности могут связываться друг с другом *отношениями*. Так, несколько сущностей типа «человек» могут быть связаны отношением «студенты группы 507». В системе подразумеваются различные *типы отношений*. В отношениях типа «группа» все члены-сущности равноправны, в отношениях типа «процесс» все члены-сущности упорядочены по дате и так далее. Система должна обеспечивать авторизированным пользователям интерфейс добавления произвольных отношений и типов отношений.

Существует специальный тип отношений — *структурные отношения*. В структурные отношения могут входить только сущности типа «веб-страница», причем любая «веб-страница» входит лишь в одно структурное отношение. Структурные отношения позволяют строить сайты на базе сущностей типа «веб-страница» в стандартном понимании термина «сайт». Так, структурному отношению ставится в соответствие сущность, которая задает шаблон всем сущностям-членам структурного отношения. Эту сущность можно включить

в новое структурное отношение и тем самым обеспечить иерархию сущностей-страниц сайта.

Еще одним объектом системы является так называемый *виджет*. Виджеты вставляются в html-поля сущностей и могут отображать другие сущности или отношения. Так, если мы хотим на одной странице сайта (т.е. сущности типа «веб-страница») сначала отобразить некоторую сущность типа «человек», скажем, профессора, а ниже отобразить все курсы, читаемые этим профессором (отношение типа «группа», связывающее его сущности-курсы), то мы в описание этой сущности-веб-страницы вставляем два виджета: виджет, отображающий сущность (человека) и виджет, отображащий отношение (список курсов).

Особенностью системы является возможность давать различные доменные имена различным сущностям-веб-страницам системы. Так, у филиала МГУ в рамках системы может быть одно доменное имя, у одного его факультета — второе, у второго — третье, и так далее. При этом, например, одна и та же сущность типа «человек» может быть представлена на всех трех сайтах тремя различными сущностями-веб-страницами при помощи виджетов. При изменении настоящей сущности изменится и ее отображение на всех этих сайтах.

В реализации системы возможность неограниченного добавления сущностей, отношений, типов сущностей и типов отношений достигается путем применения NOSQL СУБД MongoDB. В отличие от стандартных реляционных SQL СУБД, в MongoDB вместо таблиц используются коллекции, а вместо строк таблиц — документы, причем структура документа не определяется коллекцией. Другими словами, каждый документ может иметь свои уникальные поля. При этом даже поля, присутствующие лишь в некоторых документах, можно индексировать. В результате, мы получаем единое хранилище сущностей и отношений, оптимизированное для поиска в нем.

Среди других используемых в системе технологий стоит выделить серверный язык php (фреймворк YII) и клиентский язык JavaScript (библиотека jQuery).

Один из первых сайтов, реализованных в рамках системы — сайт ташкентского филиала МГУ http://uz.msu.ru

# МОДЕЛЬ ГИПЕРТЕКСТОВОЙ ОРГАНИЗАЦИИ ИНФОРМАЦИИ В ИНФОРМАЦИОННЫХ СИСТЕМАХ ДЛЯ ИСТОРИЧЕСКИХ ИССЛЕДОВАНИЙ

**Перевертень В.А. (РГГУ)**
*dpva@mail.ru*

Как отмечают специалисты по применению информационных технологий в исторической науке, создание информационных систем для исторических исследований является более сложной и специальной задачей, чем построение традиционных информационных систем. Разработка таких систем требует решения целого ряда научных и технологических задач, центральное место среди которых занимает задача организации связанной с историческими исследованиями информации.

Историко-исследовательская информация должна быть организована так, чтобы обеспечить определенную комфортность, качество и эффективность познавательно-информационной деятельности историков. Организация информации будет тем лучше, чем больше она будет соответствовать специфике исторических исследований и особенностям информационной деятельности историков как исследователей.

Ряд работ, собственные наблюдения и беседы с историками дают основания гипотетически утверждать, что с информационной т.з. в исторических исследованиях процесс познания субъектом объекта состоит в накоплении информации фрагментами и объединении и связывании образов (моделей) этих информационных фрагментов.

Фрагмент информации либо вычленяется исследователем из имеющегося в его распоряжении массива информации, либо порождается им самим. Рассматривая фрагмент по частям и воспринимая его содержание с различных т.з., с одним и тем же информационным фрагментом исследователь может связывать множество образов, поразному отражающих этот фрагмент.

Для борьбы с «проклятием размерности», которое выражается в огромном количестве информационных фрагментов и их образов, накапливающихся в процессе решения исследовательской задачи, применяются следущие два приема. Первый прием заключается в выделении части образов и объединении их в некоторые совокупности, исходя из самых произвольных соображений познающего субъекта.

Второй прием состоит в связывании образов друг с другом также по воле субъекта.

Исходя из рассмотренного представления об информационной деятельности историка, по нашему мнению, наиболее подходящей является гипертекстовая организация историко-исследовательской информации. Именно гипертекст позволяет упорядочить информацию, которую невозможно формализовать и выделить в ней необходимые для построения баз данных и баз знаний категории, у которой нет априорной структуры, с которой исследователь может работать, лишь выделяя в ней фрагменты и связывая их.

Предлагаемая нами модель гипертекстовой (гипермедиа) организации историко-исследовательской информации несколько отличается от «классического» представления о гипертексте. В ее основе лежат следующие понятия: информационный объект (ИО), образ ИО, объект гипертекста (ОГТ), ассоциация и связь.

*Информационный объект* – это фрагмент информации, который может объективно существовать или порождаться как единица информации, произвольно выделяться из имеющегося информационного массива или формироваться познающим субъектом. На содержание, форму представления, объем, структурную организацию и носитель информационного фрагмента, понимаемого как информационный объект, не накладывается никаких ограничений.

*Образ ИО* – это обозначение одного или нескольких аспектов содержания ИО, это некоторые признаки, отражающие содержание ИО, это вторичная информация, являющаяся моделью содержания ИО.

Образ ИО, дополненный уникальным идентификатором его и указателем на ИО (уникальным идентификатором ИО), к которому этот образ относится, называется *объектом гипертекста*. ОГТ выступают в качестве узлов гипертекстовой сети и играют роль «представителей» ИО в гипертексте. С одним ИО может быть связано несколько ОГТ (один – первичный, а остальные – вторичные), представляющих его с разных сторон. При такой организации сам ИО, будучи включенным в гипертекст, является пассивным компонентом гипертекста и существует независимо от него.

*Ассоциациями* мы называем группы ОГТ, которые исследователь формирует исходя из объективных свойств образов ИО или своих субъективных соображений. Каждая ассоциация имеет уникальное

имя. Один и тот же ОГТ может входить в несколько ассоциаций.

Под *связью* в предлагаемой модели гипертекстовой организации информации подразумевается симметричное бинарное отношение, устанавливаемое между двумя ОГТ относительно определенных ассоциаций, в которых связываемые ОГТ находятся. Именно в таком определении связей заключается существенное отличие нашей модели от модели «классического» гипертекста.

В формализованном виде нашу модель гипертекстовой организации информации ($HT$-модель) можно определить как пару:

$$HT = (HTS, HTP),$$

где $HTS$ – множество допустимых гипертекстовых структур;

$HTP$ – множество допустимых операций над этими структурами.

Пусть $NH$ – множество идентификаторов ОГТ, $IM$ – множество образов ИО, $ID$ – множество идентификаторов ИО, $IO$ – множество ИО, $NA$ – множество имен ассоциаций. Допустимые гипертекстовые структуры ($HTS$-структуры) определяются соответствующими отношениями, заданными на перечисленных множествах и ограничениями целостности. В предлагаемой нами модели $HTS$-структура – это кортеж:

$$HTS = (HTO, IDIO, ASS, LNK, HTSL),$$

где $HTO$ – множество ОГТ;

$IDIO$ – отношение идентификатор ИО - ИО;

$ASS$ – отношение ассоциации;

$LNK$ – множество связей;

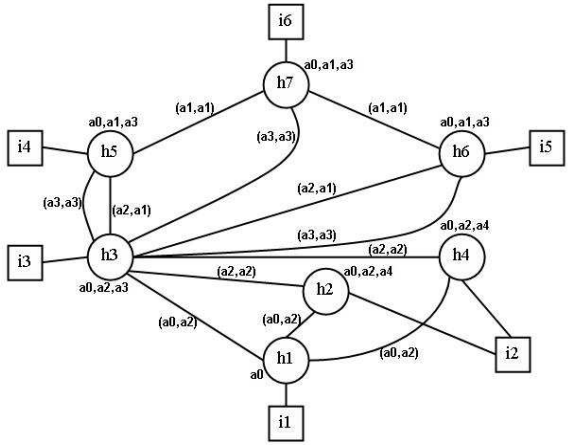
$HTSL$ – ограничения целостности гипертекстовой структуры.

$HTO \subset NH \times IM \times ID$, $IDIO \subset ID \times IO$, $ASS \subseteq NH \times NA$, $LNK \subseteq NH \times NA \times NH \times NA$, при этом $\pi_1 HTO \leftrightarrow \pi_{2,3} HTO$, $\pi_1 IDIO \leftrightarrow \pi_2 IDIO$, элементы отношения LNK представляют собой кортежи вида $(h1, a1, h2, a2)$, где $h1, h2 \in \pi_1 HTO, a1 \in ASS(h1), a2 \in ASS(h2)$, причем $h1 \neq h2$ и для каждого кортежа в $LNK$ имеется симметричный ему кортеж $(h2, a2, h1, a1)$.

Начальное состояние $HTS$-структуры соответствует утверждению: $(HTO = \emptyset)$ & $(IDIO = \emptyset)$ & $(ASS = \emptyset)$ & $(LNK = \emptyset)$. В состояниях, отличных от начального, она должна удовлетворять ограничениям целостности $HTSL$, которые заключаются в утверждении: $(\pi_1 IDIO = \pi_3 HTO)$ & $(\pi_1 ASS = \pi_1 HTO)$ & $(\pi_{1,2} LNK \subseteq ASS)$.

Наглядно $HTS$-структуру можно представить в виде графа, пример которого показан ниже (h1-h7 – идентификаторы ОГТ, i1-i6 –

идентификаторы ИО, a0-a4 – имена ассоциаций):




В множество операций $HTP$ входит 15 операций трех типов:

операции формирования гипертекста, которыми определяются изменения состояния $HTS$-структуры;

операции выборки элементов гипертекстовой структуры, предназначенные для отбора тех ее элементов, с участием которых будут выполняться некоторые операции;

операции выделения части гипертекстовой структуры в соответствии с заданными условиями.

Примером информационной системы для исторических исследований, в которой организация информации основывается на представленной выше $HT$-модели, может служить созданная автором система «Просис» [1, 2].

**Литература**

1. Гутнов Д .А., Перевертень В. А. Российские историки XVIII - начала XX вв.: проект и информационная система // Круг идей: новое в исторической информатике. Труды I конференции Ассоциации "История и компьютер". – М., 1994, с. 39-50.

2. Гутнов Д. А., Перевертень В. А. Просопографическая информационная система «Просис»: версия 2.0 // Информационный бюллетень Ассоциации "История и компьютер". – М., 1994, N 10, с. 17-18.

# МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ СУБЪЕКТИВНЫХ СУЖДЕНИЙ МОДЕЛЬЕРА-ИССЛЕДОВАТЕЛЯ О МОДЕЛИ ОБЪЕКТА ИССЛЕДОВАНИЯ

**Пытьев Ю.П. (МГУ им. М. В. Ломоносова)**
*yuri.pytyev@gmail.com*

**Аннотация**

Рассмотрены методы математического моделирования неполного и недостоверного знания модели $M(x)$ объекта исследования, выраженного в форме субъективных суждений модельера-исследователя (м.-и.) о возможных значениях неизвестного параметра $x \in X$. Математическая модель «неизвестного параметра» определена как неопределенный элемент (н. э.) $\widetilde{x}$, характеризующий (как неопределенная высказывательная переменная) субъективные суждения м.-и. об истинности каждого значения $x \in X$ значениями мер правдоподобия $\mathrm{P}l^{\widetilde{x}}$ равенства $\widetilde{x} = x$ и доверия $\mathrm{Be}l^{\widetilde{x}}$ неравенства $\widetilde{x} \neq x$.

**Предисловие.** Вероятностные и возможностные методы применяются при моделировании многих аспектов как неясности и неопределенности, отражающих неполноту и недостоверность информации, так и случайности, нечеткости и неточности, относящихся к её содержанию. Модель случайности и нечеткости обычно считается вероятностной или возможностной, а неясность и неопределенность ассоциируются с неполным знанием последней, но «моделируются», как правило, вербально [1, 2, 3]. Обычно «неполное знание» вероятностной $(\Omega, \mathcal{A}, \mathrm{Pr}(\cdot; x))$ или возможностной $(\Omega, \mathcal{P}(X), \mathrm{P}(\cdot; x))$ моделей обусловлено их зависимостью от *неизвестного* параметра $x \in X$. В научной, инженерной и прочей исследовательской и творческой деятельности невозможно исключить использование неполной, противоречивой и недостоверной информации, ассоциированной с опытными фактами, с практикой их использования и с полученными знаниями. В работе рассмотрен метод математического моделирования подобной информации о возможных значениях неизвестного параметра $x \in X$, выраженной в форме субъективных суждений [4, 5].

Существо метода иллюстрирует конструкция *неопределенного случайного элемента*, заданного на произведении пространств: вероятностного $(\Omega, \mathcal{A}, \mathrm{Pr}(\cdot; x))$, известного с точностью до значения $x \in X$ и *моделирующего неопределенный объект исследования*, и пространства $(X, \mathcal{P}(X), \mathrm{P}l^{\widetilde{x}}, \mathrm{Be}l^{\widetilde{x}})$ с мерами правдоподобия $\mathrm{P}l^{\widetilde{x}}$ и доверия $\mathrm{Be}l^{\widetilde{x}}$,

моделирующего неопределенный элемент (н. э.) $\widetilde{x}$, характеризующий субъективные суждения исследователя об истинности каждого значения параметра $x \in X$ значениями мер правдоподобия $\mathrm{Pl}^{\widetilde{x}}(\widetilde{x} = x)$ и доверия $\mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \neq x)$.

В работе показано, как такая модель «н. э. $\widetilde{x}$» позволяет *вычислить* правдоподобие и доверие *любых* суждений исследователя о *любых свойствах объекта*, обусловленных его моделью $(\Omega, \mathcal{A}, \Pr(\cdot; x))$, $x \in X$.

**Неопределенный элемент. Меры правдоподобия** $\mathrm{Pl}$ **и доверия** $\mathrm{Bel}$**.** Пусть семейство моделей $M(x)$, $x \in X$, — *неопределенная модель* объекта исследования. Модель «неизвестного параметра» $x \in X$ — *неопределенный элемент* (н. э.) $\widetilde{x}$, характеризующий субъективные суждения исследователя об истинности каждого значения $x \in X$ *значениями мер правдоподобия* $\mathrm{Pl}^{\widetilde{x}}(\widetilde{x} = x)$ и *доверия* $\mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \neq x)$.

В модели $(X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$ н. э. $\widetilde{x}$ меры правдоподобия $\mathrm{Pl}^{\widetilde{x}}(\cdot) : \mathcal{P}(X) \to \mathcal{L}$ и доверия $\mathrm{Bel}^{\widetilde{x}}(\cdot) : \mathcal{P}(X) \to \widehat{\mathcal{L}}$, где $\mathcal{L}$ и $\widehat{\mathcal{L}}$ суть шкалы их значений, определены равенствами

$$\mathrm{Pl}^{\widetilde{x}}(E) \stackrel{\mathrm{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\widetilde{x} \in E) = \sup_{x \in E} t^{\widetilde{x}}(x),\ \mathrm{Bel}^{\widetilde{x}}(E) \stackrel{\mathrm{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \in E) = \\ = \inf_{x \in X \setminus E} s^{\widetilde{x}}(x), \tag{1}$$

в которых $E \in \mathcal{P}(X)$ и

$$t^{\widetilde{x}}(x) \stackrel{\mathrm{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\widetilde{x} = x),\ s^{\widetilde{x}}(x) \stackrel{\mathrm{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \neq x). \tag{2}$$

— значения правдоподобия равенства $\widetilde{x} = x$ и доверия неравенства $\widetilde{x} \neq x$, $x \in X$. Функции $t^{\widetilde{x}}(\cdot) : X \to [0, 1]$ и $s^{\widetilde{x}}(\cdot) : X \to [0, 1]$ — *распределения правдоподобий и доверий значений* н. э. $\widetilde{x}$. Определения (1), (2) и их содержательная интерпретация суть следствия условий:

• Модельер-исследователь *может*, основываясь на своих неполных и недостоверных априорных знаниях свойств объекта, считать $x \in X$ значениями н. э. $\widetilde{x}$ и *предложить его модель* $(X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$, указав в (2), насколько, по его мнению, *относительно правдоподобны* равенства $\widetilde{x} = x$, $x \in X$, и насколько *следует относительно доверять* неравенствам $\widetilde{x} \neq x$, $x \in X$, где «относительно» означает, что в $(X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$

1) *численные значения* $\mathrm{Pl}^{\widetilde{x}}(E)$ и $\mathrm{Bel}^{\widetilde{x}}(E)$, $E \in \mathcal{P}(X)$, в (1), *отличные от нуля и единицы, не могут быть содержательно истолкованы*, а *существенна лишь их упорядоченность*,

2) меры $\mathrm{Pl}^{\widetilde{x}}(\cdot)$ и $\mathrm{Pl}'^{\widetilde{x}}(\cdot)$ ($\mathrm{Bel}^{\widetilde{x}}(\cdot)$ и $\mathrm{Bel}'^{\widetilde{x}}(\cdot)$) считаются эквивалентными, если $\exists\gamma(\cdot) \in \Gamma\ \forall E \in \mathcal{P}(X)\ \gamma(\mathrm{Pl}^{\widetilde{x}}(E)) = \mathrm{Pl}'^{\widetilde{x}}(E)$ ($\exists\gamma(\cdot) \in \Gamma\ \forall E \in \mathcal{P}(X)\ \gamma(\mathrm{Bel}^{\widetilde{x}}(E)) = \mathrm{Bel}'^{\widetilde{x}}(E)$), где $\Gamma$ — группа непрерывных, строго монотонных функций $\gamma(\cdot) : [0,1] \to [0,1]$, $\gamma(0) = 0$, $\gamma(1) = 1$, с групповой операцией «$\circ$», $\gamma \circ \gamma'(a) \stackrel{\text{def}}{=} \gamma(\gamma'(a))$, $a \in [0,1]$.

Условия 1), 2) в терминах свойств шкал $\mathcal{L}$ и $\widehat{\mathcal{L}}$, означают, что

• класс $\Gamma$ определяет *группу автоморфизмов шкал* $\mathcal{L} = ([0,1], \leqslant,\ +, \times) \equiv ([0,1], \leqslant, \max, \min)$ и $\widehat{\mathcal{L}} = ([0,1], \widehat{\leqslant}, \widehat{+}, \widehat{\times}) \equiv ([0,1], \widehat{\leqslant}, \min, \max)$ значений мер $\mathrm{Pl}^{\widetilde{x}}(\cdot) : \mathcal{P}(X) \to \mathcal{L}$ и $\mathrm{Bel}^{\widetilde{x}}(\cdot) : \mathcal{P}(X) \to \widehat{\mathcal{L}}$, то есть

$\forall\gamma(\cdot) \in \Gamma$ в шкалах $\gamma\,\mathcal{L}$ и $\gamma\,\widehat{\mathcal{L}}$: $\gamma([0,1]) = [0,1]$, $\forall a, b \in [0,1]$ $a \leqslant b \Leftrightarrow \gamma(a) \leqslant \gamma(b)$, $a \widehat{\leqslant} b \Leftrightarrow \gamma(a) \widehat{\leqslant} \gamma(b)$, и $\gamma(a * b) = \gamma(a) * \gamma(b)$, где $*$ — символ любой из *операций*: сложения $+$, $\widehat{+}$ и умножения $\times$, $\widehat{\times}$.

Равенства $a + b = a \widehat{\times} b = \max\{a, b\}$, $a \times b = a \widehat{+} b = \min\{a, b\}$, $a, b \in [0,1]$, следуют из требований [3] *непрерывности* и коммутативности операции $*$ *как отображения* $[0,1]^2 \to [0,1]$ и *свойств нейтральных элементов* 0 и 1 шкал $\mathcal{L}$ и $\widehat{\mathcal{L}}$ и группы $\Gamma$: $\forall a \in [0,1]$ $a + 0 = a \widehat{\times} 0 = a \times 1 = a \widehat{+} 1 = a$, $a + 1 = a \widehat{\times} 1 = 1$ и $a \times 0 = a \widehat{+} 0 = 0$.

Поскольку шкалы $\gamma\,\mathcal{L}$, $\gamma \in \Gamma$, и $\gamma\,\widehat{\mathcal{L}}$, $\gamma \in \Gamma$, *изоморфны*, исследователи могут формулировать модели н. э. $\widetilde{x}$ в *произвольных* шкалах $\mathcal{L}$, $\widehat{\mathcal{L}}$. Будучи сформулированными в шкалах $\mathcal{L}'$, $\widehat{\mathcal{L}}'$ и $\mathcal{L}''$, $\widehat{\mathcal{L}}''$, модели считаются *эквивалентными*, если *существуют* шкалы $\mathcal{L} = \gamma'\mathcal{L}' = \gamma''\mathcal{L}''$ и $\widehat{\mathcal{L}} = \widehat{\gamma}'\widehat{\mathcal{L}}' = \widehat{\gamma}''\widehat{\mathcal{L}}''$, $\gamma', \gamma'', \widehat{\gamma}', \widehat{\gamma}'' \in \Gamma$, в которых их формулировки совпадают, а *содержательно истолкованы* могут быть только те, формулировки которых *не зависят* от выбора шкал $\mathcal{L}$, $\widehat{\mathcal{L}}$, т. е. *одинаковы для всех исследователей*.

В терминах операций сложения и умножения в шкалах $\mathcal{L}$ и $\widehat{\mathcal{L}}$ $\forall E \in \mathcal{P}(X)$ $\mathrm{Pl}^{\widetilde{x}}(E) \stackrel{\text{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\widetilde{x} \in E) = \underset{x \in E}{+} t^{\widetilde{x}}(x)$, $\mathrm{Bel}^{\widetilde{x}}(E) \stackrel{\text{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \in E) = \underset{x \in X \setminus E}{\widehat{+}} s^{\widetilde{x}}(x)$, причем для $E = X$ и $E = \varnothing$ принимаются независящие от выбора шкал $\mathcal{L}$ и $\widehat{\mathcal{L}}$ условия $\mathrm{Pl}^{\widetilde{x}}(X) = \underset{x \in X}{+} t^{\widetilde{x}}(x) = 1$, $\mathrm{Bel}^{\widetilde{x}}(\varnothing) = \underset{x \in X}{\widehat{+}} s^{\widetilde{x}}(x) = 0$, означающие, что среди значений $x \in X$

есть истинное, определяющее модель объекта исследования.

Согласно (1), (2) если $\varphi(\cdot) : X \to Y$, $\varphi^{-1}(Y) = X$, — функция, задающая н. э. $\widetilde{y} = \varphi(\widetilde{x})$, то пространство $(Y, \mathcal{P}(Y), \mathrm{Pl}^{\widetilde{y}}, \mathrm{Bel}^{\widetilde{y}})$ — модель $\widetilde{y}$, в которой

$$\mathrm{Pl}^{\widetilde{y}}(A) = \mathrm{Pl}^{\widetilde{x}}(\varphi(\widetilde{x}) \in A) = \sup_{y \in A} t^{\widetilde{y}}(y), \tag{3}$$

$$\mathrm{Bel}^{\widetilde{y}}(A) = \mathrm{Bel}^{\widetilde{x}}(\varphi(\widetilde{x}) \in A) = \inf_{y \in Y \setminus A} s^{\widetilde{y}}(y),\ A \in \mathcal{P}(Y),$$

где

$$t^{\widetilde{y}}(y) \stackrel{\mathrm{def}}{=} \mathrm{Pl}^{\widetilde{y}}(\widetilde{y} = y) = \mathrm{Pl}^{\widetilde{x}}(\varphi(\widetilde{x}) = y) = \sup_{\substack{x \in X \\ \varphi(x) = y}} t^{\widetilde{x}}(x), \tag{4}$$

$$s^{\widetilde{y}}(y) \stackrel{\mathrm{def}}{=} \mathrm{Bel}^{\widetilde{y}}(\widetilde{y} \neq y) = \mathrm{Bel}^{\widetilde{x}}(\varphi(\widetilde{x}) \neq y) = \inf_{\substack{x \in X, \\ \varphi(x) = y}} s^{\widetilde{x}}(x).$$

Заметим, что исследователь *в любом случае может* считать $x \in X$ значениями н. э. $\widetilde{x}$ и *предложить* его модель $(X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$, ибо может воспользоваться моделями «полного знания» и «полного незнания» модели $M(X)$, $x \in X$, и свойств объекта.

Модель «полного незнания» модели $M(x)$, $x \in X$, и свойств объекта определяется распределениями: $t^{\widetilde{x}}(x) = 1$, $x \in X$, (все значения н. э. $\widetilde{x}$ равноправдоподобны), $\sup\limits_{x \in X} t^{\widetilde{x}}(x) = 1$, и $s^{\widetilde{x}}(x) = 0$, $x \in X$, (любому неравенству $\widetilde{x} \neq x$, $x \in X$, доверять нельзя), $\inf\limits_{x \in X} s^{\widetilde{x}}(x) = 0$. В этих случаях такие же распределения в (4) будет иметь любая функция $\varphi(\widetilde{x}) = \widetilde{y}$, $\varphi^{-1}(Y) = X$: $t^{\widetilde{y}}(y) = 1$, $s^{\widetilde{y}}(y) = 0$, $y \in Y$.

Модель «полного знания» модели объекта определяется распределениями: $t^{\widetilde{x}}(x) \stackrel{\mathrm{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\widetilde{x} = x) = \begin{cases} 1, & x = x_0, \\ 0, & x \neq x_0, \end{cases} x \in X$, ($x_0$ — единственное правдоподобное значение параметра) и $s^{\widetilde{x}}(x) \stackrel{\mathrm{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \neq x) = \begin{cases} 1, & x \neq x_0, \\ 0, & x = x_0, \end{cases} x \in X$, ($x_0$ — единственное значение параметра, при котором неравенству $\widetilde{x} \neq x_0$ доверять нельзя), а в (4) в этом случае $t^{\widetilde{y}}(y) = \begin{cases} 1, & y = y_0, \\ 0, & y \neq y_0, \end{cases} y \in Y$, $s^{\widetilde{y}}(y) = \begin{cases} 1, & y \neq y_0, \\ 0, & y = y_0, \end{cases} y \in Y$, где $y_0 = \varphi(x_0)$.

Разумеется, модели «полного незнания» и «полного знания» не

зависят от выбора шкал $\gamma\mathcal{L}$ и $\widehat{\gamma}\widehat{\mathcal{L}}$, $\gamma, \widehat{\gamma} \in \Gamma$, т. е. являются таковыми для любого исследователя.

**Неопределенный элемент как неопределенная высказывательная переменная.** Н. э. $\widetilde{x}$ моделирует *высказывания* исследователя о его значениях, истинность и ложность которых не абсолютны. Такая интерпретация н. э. основана на *теоретико-множественном представлении логики высказываний*, согласно которому в $(X, \mathcal{P}(X), \mathrm{P}l^{\widetilde{x}}, \mathrm{Be}l^{\widetilde{x}})$ $X$ — множество элементарных высказываний (э. в.), любое высказывание $a$ взаимно однозначно представлено множеством $A \in \mathcal{P}(X)$ э. в. $x \in X$, каждое из которых влечет $a$: $a \leftrightarrow A = \bigcup\limits_{\substack{x \in X \\ x \to a}} \{x\} \equiv \{x \in X,\ x \to a\}$, где $\leftrightarrow$ и $\to$ суть символы взаимно однозначного соответствия и логической импликации. Каждое э. в. $x \in X$ представлено в $X$ одноточечным множеством $\{x\}$, $x \leftrightarrow \{x\}$, и выделено среди всех высказываний условием: любое э. в. $x \in X$ не следует ни из какого высказывания, кроме $x$ и всегда ложного высказывания $\mathbf{0}$. При этом если $a \leftrightarrow A$, $b \leftrightarrow B$, то $a \,\&\, b \leftrightarrow A \cap B$, $a \vee b \leftrightarrow A \cup B$, $\neg a \leftrightarrow X \setminus A$, $a \to b \equiv (\neg a) \vee b \leftrightarrow (X \setminus A) \cup B$, $\mathbf{1} \leftrightarrow X$, $\mathbf{0} \leftrightarrow \varnothing$.

В таком представлении $t^{\widetilde{x}}(x) = \mathrm{P}l^{\widetilde{x}}(\widetilde{x} = x)$ $(\mathrm{P}l^{\widetilde{x}}(\widetilde{x} \in E))$ — правдоподобие истинности неопределенного высказывания (н. в.) $x \leftrightarrow \{x\}$ $(e \leftrightarrow E)$, согласно которому $\widetilde{x} = x$ $(\widetilde{x} \in E)$. Соответственно $s^{\widetilde{x}}(x) = \mathrm{Be}l^{\widetilde{x}}(\widetilde{x} \neq x)$ $(\mathrm{Be}l^{\widetilde{x}}(\widetilde{x} \in E))$ — доверие истинности н. в. $\neg x \leftrightarrow X \setminus \{x\}$ $(e \leftrightarrow E)$, согласно которому $\widetilde{x} \neq x$ $(\widetilde{x} \in E)$, $x \in X$.

Предложив модель $(X, \mathcal{P}(X), \mathrm{P}l^{\widetilde{x}}, \mathrm{Be}l^{\widetilde{x}})$ н. э. $\widetilde{x}$, исследователь согласно (3), (4) может вычислять правдоподобия и доверия истинности *любых* н. в. о значениях *любых* характеристик объекта как функций н. э. $\widetilde{x}$. Пусть, например, $M(x)$, $x \in X$, — семейство произвольных моделей, среди которых есть модель $M(x_0)$ объекта, но $x_0$ неизвестно. Если $(\Omega, \mathcal{P}(X), \mathrm{P}l^{\widetilde{x}}, \mathrm{Be}l^{\widetilde{x}})$ — его модель н. э. $\widetilde{x}$, то $M(\widetilde{x})$ — неопределенная модель объекта, и для любой неопределенной характеристики объекта $\varphi(\widetilde{x}) \stackrel{\text{def}}{=} \mathbf{C}(M(\widetilde{x}))$ правдоподобие и доверие истинности высказываний, согласно которым $\varphi(\widetilde{x}) = y$ и $\varphi(\widetilde{x}) \neq y$, $y \in Y$, определены в (4).

Далее $M(x) = (\Omega, \mathcal{A}, \mathrm{Pr}(\cdot; x))$, $x \in X$, характеристики $C(M(\widetilde{x}))$ модели $M(\widetilde{x})$ суть неопределенные вероятности $\widetilde{\mathrm{Pr}}(A) \stackrel{\text{def}}{=} \mathrm{Pr}(A; \widetilde{x})$, $A \in \mathcal{A}$, неопределенные числовые характеристики случайных вели-

чин, заданных на $(\Omega, \mathcal{A}, \widetilde{\Pr})$ и т. п.

**Неопределенные случайный элемент, вероятность и математическое ожидание.** Для пары пространств: вероятностного $(Y, \mathcal{B}, \Pr^{\eta})$ и с правдоподобием и доверием $(X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$ рассмотрим отображение $q(\cdot,\cdot) : (Y, \mathcal{B}) \times (X, \mathcal{P}(X)) \to (\Omega, \mathcal{A})$, $\mathcal{B}, \mathcal{A}$-измеримое при каждом $x \in X$: $\forall\, x \in X\ \forall\, A \in \mathcal{A}\ \{y \in Y,\ q(y,x) \in A\} \in \mathcal{B}$ и $\forall\, x \in X\ q^{-1}(\Omega, x) = Y$.

**Определение.** Функцию $\widetilde{\xi} = q(\eta, \widetilde{x})$ случайного $\eta$ и неопределенного $\widetilde{x}$ элементов назовем *неопределенным случайным элементом, заданным на произведении пространств* $(Y, \mathcal{B}, \Pr^{\eta}) \times (X, \mathcal{P}(X), \mathrm{Pl}^{\widetilde{x}}, \mathrm{Bel}^{\widetilde{x}})$ *и принимающим значения в* $(\Omega, \mathcal{A})$.

При $\widetilde{x} = x$ $\widetilde{\xi}\Big|_{\widetilde{x}=x} \stackrel{\text{def}}{=} \xi(x) = q(\eta, x)$ — случайный элемент, определенный на $(Y, \mathcal{B}, \Pr^{\eta})$ со значениями в $(\Omega, \mathcal{A}, \Pr(\cdot; x))$, $x \in X$, где

$$\Pr(A; x) \stackrel{\text{def}}{=} \Pr^{\xi(x)}(\xi(x) \in A) = \Pr^{\eta}(q(\eta, x) \in A) = \int\limits_{\substack{y \in Y,\\ q(y,x) \in A}} \Pr^{\eta}(dy),\ A \in \mathcal{A},$$

— вероятность, определяющая семейство вероятностных пространств $(\Omega, \mathcal{A},\ \Pr(\cdot; x))$, $x \in X$, *неопределенную вероятность* $\widetilde{\Pr}(A) \stackrel{\text{def}}{=} \Pr(A; \widetilde{x})$, $A \in \mathcal{A}$, и неопределенное вероятностное пространство $(\Omega, \mathcal{A}, \widetilde{\Pr})$ как неопределенную модель объекта исследования.

Согласно (4) $t^{\widetilde{\Pr}(A)}(\mathrm{pr}) \stackrel{\text{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\Pr(A; \widetilde{x}) = \mathrm{pr}) = \sup\{t^{\widetilde{x}}(x) \mid x \in X,\ \Pr(A; x) = \mathrm{pr}\}$ — правдоподобие истинности н. в., согласно которому неопределенная вероятность $\widetilde{\Pr}(A)$ события $A$ равна pr. Соответственно $s^{\widetilde{\Pr}(A)}(\mathrm{pr}) \stackrel{\text{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\Pr(A; \widetilde{x}) \neq \mathrm{pr}) = \inf\{s^{\widetilde{x}}(x) \mid x \in X,\ \Pr(A, x) = \mathrm{pr}\}$ — доверие истинности н. в., согласно которому $\widetilde{\Pr}(A) \neq \mathrm{pr}$, $\mathrm{pr} \in [0, 1]$.

Рассмотрим, например, семейство вероятностей $\Pr(\cdot; x) : \mathcal{P}(\Omega) \to [0, 1]$, $x \in X = (0, \infty)$, где $\Omega = \{\omega_1, \omega_2, \ldots\}$,

$$\Pr(\{\omega_i\}; x) \stackrel{\text{def}}{=} \mathrm{pr}_i(x) = x/(1+x)^i,\ i = 1, 2, \ldots, \tag{5}$$

$x$ — значение н. э. $\widetilde{x}$, и неопределенное вероятностное пространство $(\Omega, \mathcal{P}(\Omega), \widetilde{\Pr})$, в котором $\widetilde{\Pr}(A) \stackrel{\text{def}}{=} \Pr(A; \widetilde{x})$, $A \in \mathcal{P}(\Omega)$, как неопределенную модель объекта.

Правдоподобие $\mathrm{Pl}^{\widetilde{x}}(\forall\, i \in \{1, 2, \ldots\}\ \widetilde{\mathrm{pr}}_i = \mathrm{pr}_i)$ н. в., согласно которому $\forall\, i = 1, 2, \ldots\ \widetilde{\mathrm{pr}}_i \stackrel{\text{def}}{=} \mathrm{pr}_i(\widetilde{x}) = \mathrm{pr}_i$, не равно нулю лишь на

семействе $\mathrm{pr}_i = \mathrm{pr}_i(y) = y/(1+y)^i$, $i = 1, 2, \ldots$, $y \in (0, \infty)$, поэтому $\mathrm{Pl}^{\widetilde{x}}(\forall\, i \in \{1, 2, \ldots\}\ \widetilde{\mathrm{pr}}_i = \mathrm{pr}_i(y)) = \mathrm{Pl}^{\widetilde{x}}(\widetilde{x} = y) = t^{\widetilde{x}}(y)$ и, соответственно, $\mathrm{Bel}^{\widetilde{x}}(\exists\, i \in \{1, 2, \ldots\}\ \widetilde{\mathrm{pr}}_i \neq \mathrm{pr}_i(y)) = \mathrm{Bel}^{\widetilde{x}}(\widetilde{x} \neq y) = s^{\widetilde{x}}(y)$ суть правдоподобие н. в., согласно которому $\widetilde{\mathrm{Pr}}(\cdot) = \mathrm{Pr}(\cdot; y)$, и доверие н. в., согласно которому $\widetilde{\mathrm{Pr}}(\cdot) \neq \mathrm{Pr}(\cdot; y)$, $y \in (0, \infty)$.

Поэтому $t^{\widetilde{x}}(y)$ — *правдоподобие истинности н. в., согласно которому* $(\Omega, \mathcal{P}(\Omega), \mathrm{Pr}(\cdot; y))$ *есть модель объекта*, $s^{\widetilde{x}}(y)$ — *доверие истинности н. в., согласно которому* $(\Omega, \mathcal{P}(\Omega), \mathrm{Pr}(\cdot; y))$ *не есть его модель*, $y \in (0, \infty)$.

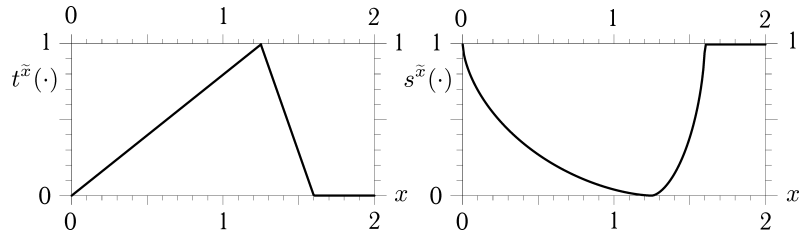



Рис. 1: Графики распределений $t^{\widetilde{x}}(x)$ и $s^{\widetilde{x}}(x)$, $x \in X = [0, 2]$, (2), предложенных исследователем.

Рассмотрим неопределенное математическое ожидание $\widetilde{\mathsf{E}}\xi_\lambda \stackrel{\mathrm{def}}{=} \mathsf{E}_{\widetilde{x}}\xi_\lambda(\cdot)$ случайной величины $\xi_\lambda(\cdot) : \Omega \to [0, 1]$, зависящей от $\lambda > 0$, заданной равенствами $\xi_\lambda(\omega_i) = \lambda^{i-1}e^{-\lambda}/(i-1)!$, $i = 1, 2, \ldots$ на семействе $(\Omega, \mathcal{P}(\Omega), \mathrm{Pr}(\cdot; x))$, $x \in X$. Так как согласно (5) $\mathsf{E}_x\xi_\lambda(\cdot) = \sum\limits_{i=1}^{\infty} \frac{\lambda^{i-1}}{(i-1)!}e^{-\lambda}\frac{x}{(x+1)^i} = \frac{x}{x+1}\exp\left(-\frac{\lambda x}{x+1}\right)$, $x \in X$, то $t^{\widetilde{\mathsf{E}}\xi_\lambda}(m) \stackrel{\mathrm{def}}{=} \mathrm{Pl}^{\widetilde{x}}(\mathsf{E}_{\widetilde{x}}\xi_\lambda(\cdot) = m) = \sup\{t^{\widetilde{x}}(x) \mid x \in (0, \infty), x(x+1)^{-1}\exp(-\lambda x(x+1)^{-1}) = m\}$, $m \in (0, \infty)$, — распределение правдоподобий н. в., согласно которым $\widetilde{\mathsf{E}}\xi_\lambda = m$, $s^{\widetilde{\mathsf{E}}\xi_\lambda}(m) \stackrel{\mathrm{def}}{=} \mathrm{Bel}^{\widetilde{x}}(\mathsf{E}_{\widetilde{x}}\xi_\lambda(\cdot) \neq m) = \inf\{s^{\widetilde{x}}(x) \mid x \in (0, \infty),\ x(x+1)^{-1}\exp(-\lambda x(x+1)^{-1}) = m\}$, $m \in (0, \infty)$, — распределение доверий н. в., согласно которым $\widetilde{\mathsf{E}}\xi_\lambda \neq m$. см. рис. 1, 2.

**Оптимальные значения н. э.** Иногда н. э., в частности, $\widetilde{\mathsf{E}}\xi_\lambda$, удобнее охарактеризовать не распределениями $t^{\widetilde{\mathsf{E}}\xi_\lambda}(\cdot)$ и $s^{\widetilde{\mathsf{E}}\xi_\lambda}(\cdot)$, а указав его, в известном смысле, *оптимальное значение*. В таком случае исследователь должен построить семейство пространств $(\mathrm{L}, \mathcal{P}(\mathrm{L}), \mathrm{Pl}^{\widetilde{l}}(\cdot|m, m'), \mathrm{Bel}^{\widetilde{l}}(\cdot|m, m'))$, $(m, m') \in \in (0, \infty)^2$, в котором $m$ — значение $\widetilde{\mathsf{E}}\xi_\lambda$, $m'$ — его оценка, L — пространство *элементарных потерь*,

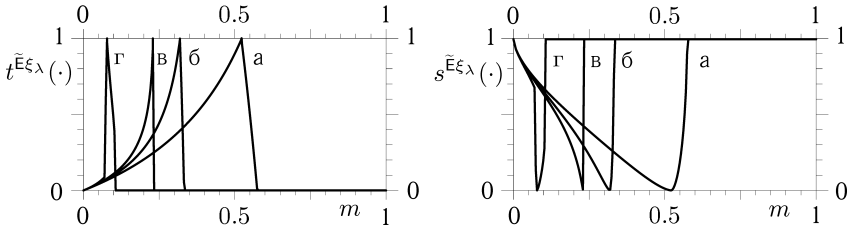



Рис. 2: Графики распределений $t^{\widetilde{\mathsf{E}}\xi_\lambda}(m)$ и $s^{\widetilde{\mathsf{E}}\xi_\lambda}(m)$, $m \in (0,\infty)$, для $\lambda = 0.11$ (а), $\lambda = 1$ (б), $\lambda = 1.58$ (в), $\lambda = 3.55$ (г).

$\mathcal{P}(\mathrm{L})$ — класс всех подмножеств L, $\mathrm{P}\widetilde{l^l}(\cdot|\cdot,\cdot)$, $\mathrm{Be}\widetilde{l^l}(\cdot|\cdot,\cdot)$ суть переходные правдоподобие и доверие для пространств $((0,\infty)^2, \mathcal{P}((0,\infty)^2))$, $(\mathrm{L}, \mathcal{P}(\mathrm{L}))$, и для каждого $\lambda \in (0,\infty)$ указать множество $V(\lambda) \in \mathcal{P}(\mathrm{L})$ *существенных* для него элементарных потерь.

Пусть $t^{\widetilde{l}}(l|m,m')$, $l \in L$, и $s^{\widetilde{l}}(l|m,m')$, $l \in L$, распределения $\mathrm{P}\widetilde{l^l}(\cdot|m,m')$ и $\mathrm{Be}\widetilde{l^l}(\cdot|m,m')$, $m,m' \in (0,\infty)$. Тогда $\mathrm{P}\widetilde{l^l}(V(\lambda)|m,m') = \sup\limits_{l\in V(\lambda)} t^{\widetilde{l}}(l|m,m') \stackrel{\mathrm{def}}{=} \mathrm{p}ll_{m,m'}$ и $\mathrm{Be}\widetilde{l^l}(V(\lambda)|m,m') = \inf\limits_{l\in \mathrm{L}\setminus V(\lambda)} s^{\widetilde{l}}(l|m,m') \stackrel{\mathrm{def}}{=} \mathrm{be}ll_{m,m'}$ суть *правдоподобие* $\mathrm{p}ll_{m,m'}$ и *доверие* $\mathrm{be}ll_{m,m'}$ *потерь, сопутствующих решению* $\widetilde{\mathsf{E}}\xi_\lambda = m'$ исследователя о значении неопределенного математического ожидания $\widetilde{\mathsf{E}}\xi_\lambda$, *в то время как на самом деле* $\widetilde{\mathsf{E}}\xi_\lambda = m$, $m,m' \in (0,\infty)$.

Соответственно $\mathop{+}\limits_{m\in(0,\infty)} (\mathrm{p}ll_{m,m'} \times t^{\widetilde{\mathsf{E}}\xi_\lambda}(m)) \stackrel{\mathrm{def}}{=} \mathrm{p}l_{t^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{p}ll_{\cdot,m'}) =$ P$l$ (потерь, сопутствующих решению $m' \in (0,\infty)$). Аналогично $\mathop{\widetilde{+}}\limits_{m\in(0,\infty)} (\mathrm{be}ll_{m,m'} \mathbin{\widetilde{\times}} s^{\widetilde{\mathsf{E}}\xi_\lambda}(m)) \stackrel{\mathrm{def}}{=} \mathrm{be}l_{s^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{be}ll_{\cdot,m'}) = \mathrm{Be}l$ (потерь, сопутствующих решению $m' \in (0,\infty)$).

Исследователь принимает оптимальные решения $m'^*$ и $m'_*$ как *минимизирующие правдоподобие* $\mathrm{p}l_{t^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{p}ll_{\cdot,m'})$ *и доверие* $\mathrm{be}l_{s^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{be}ll_{\cdot,m'})$ *потерь* на множестве решений $m' \in (0,\infty)$:

$$\mathrm{p}l_{t^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{p}ll_{\cdot,m'^*}) \stackrel{\mathrm{def}}{=} \sup_{m\in(0,\infty)} \min\{\mathrm{p}ll_{m,m'^*}, t^{\widetilde{\mathsf{E}}\xi_\lambda}(m)\} \equiv$$

$$\equiv \mathop{+}_{m\in(0,\infty)} (\mathrm{p}ll_{m,m'^*} \times t^{\widetilde{\mathsf{E}}\xi_\lambda}(m)) = \min_{m'\in(0,\infty)} \sup_{m\in(0,\infty)} \min\{\mathrm{p}ll_{m,m'}, t^{\widetilde{\mathsf{E}}\xi_\lambda}(m)\},$$

$$\mathrm{be}l_{s^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{be}ll_{\cdot,m'_*}) \stackrel{\mathrm{def}}{=} \inf_{m\in(0,\infty)} \max\{\mathrm{be}ll_{m,m'_*}, s^{\widetilde{\mathsf{E}}\xi_\lambda}(m)\} \equiv$$

$$\equiv \mathop{\widehat{+}}_{m\in(0,\infty)} (\mathrm{be}ll_{m,m'_*} \mathop{\widetilde{\times}} s^{\widetilde{\mathsf{E}}\xi_\lambda}(m)) = \min_{m'\in(0,\infty)} \inf_{m\in(0,\infty)} \max\{\mathrm{be}ll_{m,m'}, s^{\widetilde{\mathsf{E}}\xi_\lambda}(m)\},$$

где p$l$- и be$l$-интегралы $\mathrm{p}l_{t^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{p}ll_{\cdot,m'})$, $m'\in(0,\infty)$, и $\mathrm{be}l_{s^{\widetilde{\mathsf{E}}\xi_\lambda}}(\mathrm{be}ll_{\cdot,m'})$, $m'\in(0,\infty)$, предполагаются полунепрерывными снизу.

Автор признателен Ю. М. Нагорному, рассчитавшему зависимости на рис. 2 и подготовившему электронный вариант рукописи.

**Литература**

1. Пытьев Ю. П. Методы математического моделирования измерительно-вычислительных систем. – Москва, Физматлит, 2011, третье издание.

2. Дюбуа Д., Прад А. Теория возможностей. – М.: Радио и связь, 1990.

3. Пытьев Ю. П. Возможность как альтернатива вероятности. – М.: Физматлит, 2011 (второе издание).

4. Пытьев Ю. П. Неопределенные нечеткие модели и их применения. // Интеллектуальные системы. — 2004. — Т. 8, вып. 1–4. — С. 147–310.

5. Пытьев Ю. П. Эмпирическое восстановление мер возможности и правдоподобия возможности в моделях экспертных решений. // Автоматика и телемеханика, № 3, 2010, с. 131–146.

# МАТЕМАТИЧЕСКИЙ МЕТОД ОПРЕДЕЛЕНИЯ КАТАЛИТИЧЕСКОЙ АКТИВНОСТИ ФЕРМЕНТОВ В СЛОЖНЫХ БИОЛОГИЧЕСКИХ РАСТВОРАХ

**Садовничий В.А., Ветров Д.П., Вишневский В.В., Галатенко А.В., Галатенко В.В., Зыкова Т.В., Коршунов А.А., Лебедев А.Е., Лукашенко Т.П., Подольский В.Е., Политов А.В.**
**(ИМИСС МГУ имени М.В. Ломоносова)**
*vgalat@imscs.msu.ru*

Информация о каталитической активности ферментов в сложных биологических растворах (в частности, в крови) важна для корректной постановки медицинского диагноза и выбора правильной стратегии лечения. При наличии априорной информации о присутствующих в растворе ферментах эта информация может быть получена стандартными методами, а именно, применением ингибиторов и сведением к задаче определения каталитической активности фермента в чистом растворе (то есть в растворе, содержащем только один активный фермент). Однако интерес представляет и постановка, в которой априорных данных о присутствующих в сложном растворе ферментах нет.

Математическая часть этой задачи может быть сформулирована следующим образом. Имеется реализация функции $v(S)$, сопоставляющая каждому неотрицательному значению $S$ соответствующее ему значение функции $v$. При этом известно, что аналитически функция $v(S)$ представима в виде

$$v(S) = \sum_{n=1}^{N} \frac{v_{max}^{(n)} S}{K_M^{(n)} + S}$$

($K_M^n$ — попарно различные положительные константы, представляющие собой константы Михаэлиса входящих в раствор ферментов, $v_{max}^n$ — положительные константы, характеризующие максимальную скорость реакции для чистого раствора отдельного фермента), но параметры $N$, $K_M^{(n)}$ и $v_{max}^{(n)}$ неизвестны. Химически это соответствует модели Михаэлиса–Ментен (см., например, [1], [2]) и отсутствию взаимного влияния ферментов. Задача заключается в восстановлении по реализации функции $v(S)$ параметров $N$, $K_M^{(n)}$ и $v_{max}^{(n)}$.

Предлагаемый метод решения этой задачи состоит из трех этапов: интегральное преобразование, комплексно-аналитическое про-

должение и разложение по ортогональной системе в пространстве почти периодических функций.

Перед перечисленными содержательными шагами осуществляется переход от функции $v(S)$ к реально исследуемой функции $f(t)$. Этот переход состоит в вычитании из $v(S)$ величины $v_\infty = \lim\limits_{S\to\infty} S(t)$, умножении результата на минус единицу и переходе от переменной $S$ к переменной $t = \sqrt{S}$. В результате функция $f(t)$ имеет вид

$$\sum_{n=1}^{N} \frac{v_{max}^{(n)} K_M^{(n)}}{K_M^{(n)} + t^2}.$$

Применение к функции $f(t)$ интегрального преобразования Фурье

$$F(z) = \int\limits_{-\infty}^{+\infty} f(t) e^{-itz}\, dt$$

(см., например, [3, Гл. VIII, § 4,5]), с учетом равенства

$$\int\limits_{-\infty}^{+\infty} \frac{e^{-i\lambda x}}{x^2 + a^2}\, dx = \frac{\pi}{a} e^{-a|\lambda|}$$

(см., например, [3, Гл. VIII, § 4, п. 1, пример 3]), переводит функцию $f(t)$ в функцию

$$F(z) = \sum_{n=1}^{N} v_{max}^{(n)}\, \gamma_n\, \pi e^{-\gamma_n z},$$

где $\gamma_n = \sqrt{K_M^{(n)}}$.

Следует отметить, что вместо стандартного (экспоненциального) преобразования Фурье возможно (без изменения дальнейшей сути метода) и применение косинус-преобразования.

Далее осуществляется комплексно-аналитическое продолжение функции $F(z)$ с луча $[0; +\infty)$ действительной прямой на квадрант $\operatorname{Re} z \geqslant 0$, $\operatorname{Im} z \geqslant 0$ комплексной плоскости. Такое продолжение осуществимо в силу аналитичности функций $e^{-\gamma_n z}$ во всей комплексной плоскости. Комплексно-аналитического продолжение может быть реализовано на основе стандартных конечно-разностных

схем и определения комплексной производной или же эквивалентных комплексной дифференцируемости условий Коши–Римана: оценки производных могут осуществляться рассмотрением горизонтальных приращений, а затем, на основе рассмотрения вертикальных приращений, могут быть восстановлены значения функции на следующем уровне сетки.

После комплексно-аналитического продолжения осуществляется переход к функции действительного переменного $L(x)$: $L(x) = \operatorname{Re} F(ix)$ $(x \geqslant 0)$. Аналитически функция $L(x)$ представляет собой сумму

$$\sum_{n=1}^{N} v_{max}^{(n)} \gamma_n \pi \cos(\gamma_n x).$$

Наконец, для оценки характеристик присутствующих в растворе ферментов используются базовые факты теории почти периодических функций (см., например, [4], [5, Доп.: Почти периодические функции]).

На пространстве почти почти периодических функций определено скалярное произведение

$$(f, g) = \lim_{T \to \infty} \frac{1}{T} \int_0^T f(x) \overline{g(x)}\, dx,$$

и система $\{cos(\lambda x)\}_{\lambda > 0}$ является ортогональной относительно этого скалярного произведения.

Соответственно, для оценки количества ферментов и их характеристик функция $L(x)$ разлагается по системе $\{\cos(\lambda x)\}_{\lambda > 0}$ или, иными словами, строится функция $C(\gamma) = (L(x), \cos(\gamma x))$ $(\gamma > 0)$. Функция $C(\gamma)$ отлична от нуля лишь в точках $\gamma_n$ $(n = 1, 2, \ldots, N)$, причем

$$C(\gamma_n) = \frac{v_{max}^{(n)} \gamma_n \pi}{2}.$$

Таким образом, число слагаемых (число ферментов) $N$ совпадает с числом точек, в которых функция $C(\gamma)$ совершает скачек (принимает ненулевое значение), и каждая такая точка позволяет восстановить значение $K_M^{(n)}$ $(K_M^{(n)} = \gamma_n^2)$ и значение $v_{max}^{(n)}$ $(v_{max}^{(n)} = \frac{2C(\gamma_n)}{\pi \gamma_n})$.

Так как в приложении перебор континуального множества (проведение вычислений для всех действительных $\gamma$) неосуществим, реальные вычисления осуществляются лишь для сетки аргументов. В этом случае по данным о порядках максимально и минимально возможного значения $K_M$ для присутствующих в растворе ферментов и данным о минимально возможном различии между значениями $K_M$ для различных ферментов определяется мелкость сетки, а также параметр $T_{max}$, используемый при оценке скалярных произведений в качестве значения для $T$ (а переход к пределу при $T \to \infty$ не осуществляется). Тогда при значениях $\gamma$, близких к $\gamma_n$, у функции $C(\gamma)$ будет наблюдаться "горб" с высотой порядка

$$\frac{v_{max}^{(n)}\,\gamma_n\,\pi}{2},$$

а при значениях $\gamma$, далеких от всех $\gamma_n$, функция $C(\gamma)$ будет принимать малые значения (то есть будет наблюдаться шум сравнительно малой амплитуды). По этим данным (количество "горбов", их центры и высоты) и осуществляется оценка значений $N$, $v_{max}^{(n)}$ и $K_M^{(n)}$.

Следует отметить, что для определения каталитической активности ферментов вместо интегрального преобразования Фурье в качестве интегрального преобразования может использоваться обратное преобразование Лапласа (см., например, [3, Гл. VIII, § 6]). В этом случае интегральное преобразование осуществляется после комплексно-аналитического продолжения.

Работа выполнена в рамках Проекта по развитию пост-геномных исследований и технологий "Разработка и реализация математических методов для определения каталитической активности ферментов в сложных биологических растворах" (в соответствии с приказом ректора МГУ имени М.В. Ломоносова № 484 от 25 мая 2011 года).

**Литература**

1. Яковлев В.А. Кинетика ферментативного катализа – М., 1965.
2. Уэбб Л. Ингибиторы ферментов и метаболизма – М., 1966.
3. Колмогоров А.Н., Фомин С.В. Элементы теории функций и функционального анализа – М., 1976.
4. Левитан Б.М. Почти-периодические функции – М., 1953.
5. Демидович Б.П. Лекции по математической теории устойчивости – М., 1967.

# ПОЛУЧЕНИЕ, ОБРАБОТКА И ВОСПРОИЗВЕДЕНИЕ МЕДИЦИНСКОЙ ТАКТИЛЬНОЙ ИНФОРМАЦИИ

**Садовничий В.А., Соколов М.Э., Бармин В.В., Буданов В.М., Галатенко А.В., Галатенко В.В., Коршунов А.А., Козорезов Ю.Ю., Подольский В.Е.**
**(МГУ имени М.В. Ломоносова)**
*vgalat@imscs.msu.ru*

Наряду со зрением и слухом тактильные ощущения являются для человека одним из основных источников информации о внешнем мире. Тактильная информация существенно отличается от информации, получаемой посредством зрения и слуха. Связано это, в частности, с разным характером ощущений (тактильные ощущения являются контактными, а не дистантными).

К настоящему времени для визуальной информации (изображения, видео-поток без звука) и аудио-информации, а также для объединения этих видов информации (видео-поток со звуком) разработана и активно используется формальная база, включающая четкие стандарты и позволяющая строго говорить об обработке, хранении, воспроизведении и анализе такого рода данных и в аналоговом, и в цифровом форматах. В этой связи ограничимся лишь упоминанием цифровых стандартов JPEG/JPEG 2000 (см., например, [1]), MP3 (MPEG-1/2 Audio Layer 3, см., например, [2], [3]) и MPEG-2 (см., например, [4]). Для тактильной же информации до последнего времени не было разработано ни формальной модели, ни устройств, обеспечивающих получение и воспроизведение такого рода информации.

Однако в последние годы был разработан медицинский тактильный эндохирургический комплекс, обеспечивший возможность получения, хранения, воспроизведения и анализа медицинской тактильной информации. Основными составляющими комплекса являются тактильный механорецептор (см., например, [5]), тактильный дисплей и управляющая станция.

Тактильный механорецептор представляет собой круговой цилиндр с диаметром основания 10–20 мм, на одном из оснований которого расположена рабочая головка — электронная “таблетка”, содержащая датчики давления и обеспечивающая регулярное снятие показателей датчиков (в текущей реализации — 100 раз в секунду) и передачу этих показателей управляющей станции. Дополнительно механорецептор может быть оборудован системой, позволяющей

точно определять текущее положение и ориентацию этого прибора. Мобильная версия тактильного механорецептора предназначена для получения информации в ходе эндоскопических операций. Стационарная версия предназначена для исследования удаленных тканей. В настоящее время механорецептор успешно проходит клинические испытания.

Тактильный дисплей представляет собой устройство, позволяющее воспроизводить тактильную информацию, в частности, информацию, полученную от тактильного механорецептора. Основной частью тактильного дисплея является экран, состоящий из рабочих элементов — пикселов. Каждый рабочий элемент (независимо от других) может изменять свое текущее положение и текущую жесткость, а также получать и передавать информацию о силе воздействия (текущем давлении) на этот элемент.

Управляющая станция обеспечивает получение информации от тактильного механорецептора и тактильного дисплея, передачу команд на тактильный дисплей, хранение, обработку и анализ тактильной информации.

Медицинский тактильный эндохирургический комплекс позволяет решить ряд важных медицинских и смежных задач. Перечислим некоторые из них.

**Объективизация тактильных данных.** Переход от субъективного описания тактильных характеристик ткани (“мягкая”, “умеренно жесткая” и т.п.) к объективным данным обеспечивает возможность сопоставления результатов различных тактильных исследований, в частности, дает возможность объективно оценивать динамику изменений.

**Получение тактильной информации при проведении эндоскопических операций.** При проведении эндоскопических операций визуальной информации может оказаться недостаточно, а получение тактильной информации стандартным образом в данном случае невозможно. Особенно это актуально в грудной хирургии, так как при выполнении операции легкое спадается. В этих условиях различить патологический очаг, расположенный в глубине легочной ткани, например, опухолевый узел, бывает практически невозможно из-за его смещения.

Кроме того, исследование тканей тактильным механорецептором в ходе операции значительно превосходит по скорости срочное гисто-

логическое исследование. Последнее, к тому же, требует круглосуточной работы специальной лаборатории, что по организационным причинам оказывается неосуществимым во многих стационарах в ходе экстренных операций, проводимых в ночное время. Такая ситуация часто приводит к ошибкам в постановке диагноза и тактическим промахам, в то время как использование тактильного механорецептора делает возможным проведение исследования членом операционной бригады без привлечения других специалистов.

Таким образом, применение тактильного механорецептора интраоперационно является эффективным способом получения информации, необходимой для правильной диагностики и выработки плана дальнейшего ведения операции.

**Телемедицина.** Медицинский тактильный комплекс позволяет осуществлять передачу и воспроизведение тактильной информации по локальным и глобальным сетям в реальном масштабе времени, обеспечивая тем самым возможность удаленного консультирования во время проведения операций.

**Обучение.** Использование тактильного дисплея позволяет показывать студентам-медикам свойства тканей с различными патологиями в условиях отсутствия реальных образцов таких тканей.

**Автоматизированный анализ.** Интраоперационный автоматизированный анализ тактильных данных позволяет в реальном масштабе времени диагностировать наличие неоднородностей, выявлять ткани с заданными тактильными характеристиками. Также воспроизведение на тактильном дисплее специальным образом обработанной тактильной информации (например, информации с усиленной тактильной контрастностью) позволяет увеличить качество диагностики в сложных случаях.

Таким образом, медицинский тактильный комплекс предоставляет новые для медицины возможности, эффективно дополняя существующее оборудование.

Следует отметить, что в дальнейшем представляется естественным применять тактильный комплекс не только в медицине, но и в других приложениях. Также важной представляется естественная задача построения сравнительно универсальной модели тактильной информации, позволяющей говорить об обработке и анализе тактильных данных и при этом не ограниченной спецификой теку-

щей реализации медицинского тактильного эндохирургического комплекса.

Работа выполнена в рамках комплексного проекта 2010-218-01-345 Министерства образования и науки РФ “Организация производства медицинских и биологических устройств с тактильными возможностями”.

**Литература**

1. Taubman D.S., Marcellin M.W. JPEG2000: image compression fundamentals, standards, and practice – Norwell, Massachusetts: Kluwer Academic Publishers, 2004.

2. Brandenburg K.-H., Stoll G. et al. ISO-MPEG-1 Audio: A Generic Standard for Coding of High Quality Digital Audio // JAES, **42** (10), 1994, 780–792.

3. Information Technology: Generic coding of Moving pictures and associated audio – Audio Part – International standard ISO/IEC 13818-3, 1995.

4. Watkinson J. MPEG 2 – Oxford: Focal Press, 1999.

5. Садовничий В.А., Буданов В.М., Соколов М.Э. и др. Математические задачи и методы в тактильной диагностике – М.: МАКС Пресс, 2008.

# МАТЕМАТИЧЕСКИЕ МЕТОДЫ АВТОМАТИЗАЦИИ ТАКТИЛЬНОЙ ДИАГНОСТИКИ

**Садовничий В.А., Соколов М.Э., Ветров Д.П., Галатенко А.В., Галатенко В.В., Зыкова Т.В., Лебедев А.Е., Лукашенко Т.П., Подольский В.Е., Политов А.В.**
**(МГУ имени М.В. Ломоносова)**
*vgalat@imscs.msu.ru*

Тактильные ощущения являются важным источником информации для человека. В ряде случаев, в том числе в медицинской диагностике, именно тактильные данные являются основой, на которой строится оценка ситуации, выработка гипотез и принятие решений.

В то же время, до последнего времени, в отличие от визуальных образов, аудио- и видео-данных, тактильные данные носили исключительно субъективный характер, и модель, позволяющая строго говорить об обработке и анализе тактильных данных, отсутствовала. Однако после появления специального устройства (тактильного механорецептора), используемого в хирургии, в том числе, при проведении эндоскопических операций, появилась возможность говорить об объективных, численно измеряемых тактильных данных: результатом исследования, основанного на использовании тактильного механорецептора, является поле давлений $P(x_n, t_m)$, снимаемое с датчиков $x_n$ рабочей головки механорецептора в процессе исследования. В результате возникла задача об анализе этих данных, в первую очередь, задача автоматизации медицинской диагностики.

Неформально задача автоматизации диагностики может быть сформулирована следующим образом. Известно, какая именно ткань обследуется тактильным механорецептором и какие диагнозы возможны. Требуется на основе результатов тактильного исследования из множества возможных диагнозов выбрать наиболее вероятный (или наиболее вероятные). При этом можно считать, что для каждого возможного диагноза имеется, во-первых, экспертное описание тактильных характеристик ткани, свойственных этому заболеванию, и, во-вторых, набор образцов тканей с данным диагнозом.

Описанная задача может быть формализована следующим образом. Имеется банк данных с информацией о тактильных свойствах тканей. Более конкретно, имеется совокупность диагнозов, и для каждого варианта диагноза имеется экспертное описание свойств тканей с таким диагнозом, а также совокупность результатов об-

следований образцов тканей с таким диагнозом. Имеется результат обследования исследуемой ткани. Для каждого диагноза, данные о котором есть в банке данных, требуется указать одно или несколько чисел, характеризующих достоверность (или недостоверность) данного диагноза применительно к исследуемой ткани. В результате диагнозы упорядочиваются по убыванию достоверности, и можно увидеть, является ли какой-либо диагноз бесспорным, есть ли среди диагнозов возможные, или же ни один диагноз из банка данных не применим к исследуемой ткани.

Для эффективного применения в клинической практике метод автоматизации диагностики должен удовлетворять следующим требованиям: результаты диагностики должны быть формально обоснованы; метод должен иметь низкую вычислительную сложность, позволяющую применять его в реальном масштабе времени; метод должен быть устойчив к изменениям входных данных, вызванным технологическими погрешностями; желательна возможность сочетания экспертных знаний и экспериментальных данных (то есть результатов обследования образцов с известным диагнозом); желательно сопровождение результатов показателями достоверности.

Эти требования делают невозможным использование для автоматизации диагностики ряда стандартных методов решения задач классификации. Так, применение экспертных систем (см., например, [1], [2]) не дает возможности эффективно использовать экспериментальные данные, а также требует регулярного обновления банка правил в “ручном” режиме. Применение нейронных сетей (см., например, [3], [4]), наоборот, затрудняет использование экспертных данных, а также не удовлетворяет требованию о формальной обоснованности вывода. Классификация на основе тестов (см., например, [5]) также приводит к невозможности эффективного сочетания экспертных знаний и экспериментальных данных: при автоматическом создании набора тестов не могут быть эффективно использованы экспертные знания (а также нарушается требования о формальной обоснованности вывода), а при неавтоматическом создании набора тестов возникает сложность учета экспериментальных данных (и возникает необходимость регулярного обновления набора тестов в “ручном” режиме).

В рассматриваемой ситуации оптимальным подходом к решению задачи автоматизации диагностики представляется методика,

сочетающая в себе идеи сразу нескольких классических подходов, в первую очередь, классификации, основанной на кластерном анализе (см., например, [6], [7]), а также теории вероятностей и математической статистики (вероятностные модели).

Основные шаги разработанного метода автоматизации диагностики заключаются в следующем.

**Понижение размерности.** Осуществляется переход от непосредственных результатов тактильного исследования к совокупности из сравнительно небольшого числа характеристик, каждая из которых представляет собой функцию от результатов исследования. На характеристики налагаются требования корректной зависимости от результатов исследования и наличия естественной трактовки, понятной медикам и другим специалистам. Именно второе требование делает предлагаемый способ понижения размерности в рассматриваемой ситуации более привлекательным по сравнению с широко используемым методом главных компонент (см., например, [7]).

**Предобработка банка данных.** Результаты обследований банковских образцов ткани с известным диагнозом рассматриваются как точки пространства характеристик. Набор полученных точек пополняется точками, построенными на основе экспертных знаний (это осуществимо благодаря наличию и характеристик естественных трактовок). Для каждого диагноза на основе полученного множества точек автоматически строится функция (функции), сопоставляющая точкам пространства характеристик оценку вероятности того, что ткань, исследование которой порождает эту точку, соответствует данному диагнозу. Построение этой функции основано на определении для каждого диагноза метрики в пространстве характеристик, учитывающей особенности этого диагноза. Допускается возможность "ручной" коррекции построенной функции.

**Предсказание диагноза.** Осуществляется переход от непосредственных результатов обследования ткани с неизвестным диагнозом к точке пространства характеристик, и для этой точки (и для каждого возможного диагноза) вычисляются функции, характеризующие достоверность диагноза, после чего диагнозы упорядочиваются по убыванию достоверности. Помимо основных показателей достоверности того, что рассматриваемый диагноз применим или не применим к ткани, для каждой из используемых характеристик вычисляется показатель согласованности с типичной для этого диагноза

картиной. Совокупность таких показателей согласованности также представляет собой полезную информацию, показывающую, какие характеристики подтверждают диагноз, а какие опровергают.

Разработанный метод позволяет при настройке системы эффективно сочетать экспертные знания и экспериментальные данные; дает возможность естественной интерпретации результатов диагностики, а также обеспечивает обоснованность принятия решения; позволяет корректно обрабатывать ситуации, когда к изучаемому образцу ткани не применим ни один из диагнозов банка данных, а также ситуации, когда к изучаемому образцу применимы сразу несколько диагнозов; обладает низкой вычислительной сложностью и высокой устойчивостью. На эвристическом уровне можно также отметить, что разработанный метод автоматизации диагностики в некотором смысле близок к тому, что на самом деле делает врач.

Апробация на модельном банке данных показала эффективность разработанного метода автоматизации медицинской диагностики, осуществляемой на основе данных тактильного исследования.

Работа выполнена в рамках комплексного проекта 2010-218-01-345 Министерства образования и науки РФ “Организация производства медицинских и биологических устройств с тактильными возможностями”.

**Литература**

1. Джексон П. Введение в экспертные системы. 3-е издание – М.: Издательский дом “Вильямс”, 2001.
2. Джарратано Дж., Райли Г. Экспертные системы: принципы разработки и программирование – М.: Издательский дом “Вильямс”, 2007.
3. Уоссерман Ф. Нейрокомпьютерная техника. Теория и практика – М.: Мир, 1992.
4. Псиола В.В. Обзор основных нейросетевых моделей // Интеллектуальные системы, **4** (вып. 3-4), 1999.
5. Кудрявцев В.Б., Андреев А.Е., Гасанов Э.Э. Теория тестового распознавания – М.: Физматлит, 2007.
6. Мандель И.Д. Кластерный анализ – М.: Финансы и статистика, 1988.
7. Айвазян С.А., Бухштабер В.М., Енюков И.С., Мешалкин Л.Д. Прикладная статистика. Классификация и снижение размерности – М.: Финансы и статистика, 1989.

# АЛГОРИТМЫ УПОРЯДОЧИВАНИЯ ПЕРЕМЕННЫХ В ЛОКАЛЬНОМ ЭЛИМИНАЦИОННОМ АЛГОРИТМЕ

**Свириденко А.В., Щербина О.А.**
**(Таврический национальный университет имени В.И. Вернадского)**
*oleks.sviridenko@gmail.com, oshcherbina@gmail.com*

Для решения разреженных задач дискретной оптимизации (ДО) путем вычисления *глобальной информации* с помощью *локальных вычислений* на основе анализа окрестностей элементов задачи [1] может быть использован класс локальных элиминационных алгоритмов (ЛЭА) [2].

Анализ публикаций, посвященных этой проблеме, позволяет сделать вывод, что в настоящее время экспериментальное исследование поведения алгоритма несериального динамического программирования (НСДП) [3, 4], являющегося локальным алгоритмом элиминации переменных, не проводилось.

НСДП последовательно элиминирует переменные в порядке, задаваемом упорядочением переменных, которое существенно влияет на время счета. На практике для нахождения элиминационной последовательности переменных используются всевозможные эвристики (такие как метод минимальной степени, рекурсивного разбиения и др.).

Нерешенным является вопрос экспериментального исследования поведения алгоритма НСДП в зависимости от использования алгоритма упорядочивания переменных.

Целью настоящей работы является анализ влияния каждого из пяти алгоритмов упорядочивания переменных, оказываемое ими на время решения разреженной задачи ДО с помощью алгоритма НСДП.

Рассмотрим разреженную задачу ДО, структура которой задается графом взаимосвязей переменных [3, 4]. В графе взаимосвязей задачи ДО вершины соответствуют переменным, причем две вершины соединяются ребром, если соответствующие переменные имеются в одном и том же ограничении или/и в одном компоненте целевой функции.

Алгоритм НСДП может быть кратко записан так:

1. Перенумеровать переменные согласно порядку $\alpha$ (в данном случае $\alpha : \{x_1, \ldots, x_n\}$).

2. Для $i = 1, \ldots, n$ исключить $x_i$, изменяя граф взаимосвязей путем добавления ребер для превращения окрестности исключаемой вершины в клику.

**Алгоритм минимальной степени MD.** В алгоритме минимальной степени (minimum degree (MD)) выбирается вершина $v$ графа $G$ с минимальной степенью. Далее строится граф $G'$, получаемый путем создания клики из вершины $v$ и ее соседей с последующим удалением $v$ и инцидентных ей ребер. Рекурсивно из $G'$ с помощью эвристики создается *хордальный* суперграф $H'$. И, наконец, получают связный суперграф $H$ из $G$, путем добавления $v$ и инцидентных ей ребер из $G$ к $H'$. Будучи алгоритмом локальной минимизации, алгоритм MD не всегда дает упорядочение с минимальным пополнением для графа в целом. Для получения упорядочения при помощи алгоритма минимальной степени MD в данной работе была выбрана функция minimum_degree_ordering() из библиотеки BOOST [5].

**Алгоритм рекурсивного разбиения ND** Алгоритм рекурсивного разбиения ND [6] находит сепаратор, т.е. множество вершин $S$, разделяющее граф на две части $A$ и $B$, помещая в упорядочении последним. Алгоритм применяется рекуррентно к частям графа $A$ и $B$, пока их размеры не станут меньше, чем некоторое пороговое значение. Для вычисления упорядочения с помощью алгоритма рекурсивного разбиения ND в настоящей работе использовалась функция METIS_EdgeND() библиотеки Metis [7].

**Алгоритм MCS** В [8] был предложен алгоритм MCS (Maximum Cardinality Search – поиск по максимальной степени). MCS на некотором графе $G$ производит за время $O(n + m)$ полное упорядочение множества вершин следующим образом. Из произвольной вершины, выбирается любая, еще не пронумерованная вершина, смежная максимальному числу уже пронумерованных вершин. MCS упорядочение было получено при помощи пакета CHOMPACK [9].

**Алгоритм Minimum Fill-in** Эвристика минимального пополнения (Minimum Fill-in) [10] работает практически так же, как и описанная выше эвристика минимальной степени MD, с той лишь разницей, что здесь на каждом шаге выбирается такая вершина, чтобы число добавляемых к ней ребер, необходимых для получения клики, было минимальным.

**Алгоритм Lex-BFS** Rose, Tarjan, и Lueker [11] предложили алго-

ритм, вычисляющий хорошую элиминационную последовательность за линейное время, который был назван *лексикографическим поиском в ширину* (Lex-BFS). Суть данного метода заключается в следующем. Вершины нумеруются от $n$ до 1 (нумерация фиксирует позиции переменных) в упорядочении. Далее, для каждой вершины создается метка, содержащая множество чисел, записанных по убыванию. Таким образом вершины могут быть лексикографически упорядочены согласно их меткам.

**Описание вычислительного эксперимента**

Алгоритм НСДП реализован на языке программирования Python, который был выбран в связи с наличием в нем средств для быстрого построения и описания задач ДО, а также ввиду обширного количества имеющихся библиотек для работы с графами и алгоритмами упорядочивания.

**Описание множества тестовых задач.** Тестовые задачи ДО генерировались на основе уже существующих гиперграфов из библиотеки задач удовлетворения ограничений [12]. Для построения ограничения $i$ бралось очередное гиперребро гиперграфа из библиотеки [12], содержащее множество переменных $X_{S_i}$, входящих в строящееся ограничение. Далее с помощью процедуры, использующей датчик случайных чисел, строились коэффициенты $A_{S_i}$ при соответствующих переменных, тогда левая часть $i$-го ограничения имела вид $A_{S_i}X_{S_i}$. Правая часть $i$-го ограничения имеет вид $\sigma \sum A_{S_i}$, где $\sigma$ – случайное число из интервала (0, 1). Коэффициенты $c_j$ целевой функции $\sum_{j=1}^{n} c_j x_j \to \max$. также строились с помощью процедуры, использующей датчик случайных чисел. Далее для каждой тестовой задачи ДО строился соответствующий граф взаимосвязей. Для каждого графа взаимосвязей применялись алгоритмы упорядочивания MD, ND, MCS, MIN-FILL и LEX-BFS, после чего задачи ДО решались с помощью алгоритма НСДП с соответствующими найденными упорядочениями.

**Анализ результатов вычислительного эксперимента.** Для проведения вычислительного эксперимента были взяты пять групп тестовых задач: 'dubois', 'bridge', 'adder', 'pret' и 'NewSystem' из библиотеки [7], содержащие 33 тестовые задачи. Все вычисления были проведены на базе процессора Intel Core 2 Duo @ 2.66 GHz, 2 GB ОЗУ и операционной системы Linux, версия ядра 2.6.35-24-generic. Согласно результатам вычислительного эксперимента, на множестве

тестовых задач для алгоритма ND минимальное время работы алгоритма НСДП было достигнуто 0 раз (0%), для MD – 2 раза (6,0%), LEX-BFS – 3 раза (9,1%), MCS – 9 раз (27,3%) и MIN-FILL – 19 раз (57,6%).

**Литература**


1. Журавлев Ю. И. Избранные научные труды. – М.: Магистр, 1998.

2. Щербина О. А. Локальные элиминационные алгоритмы решения разреженных дискретных задач // Журнал вычислительной математики и математической физики. – 2008. – Т. 48, N 1. – С. 161-177.

3. Bertele U., Brioschi F. Nonserial Dynamic Programming. – New York: Academic Press, 1972.

4. Щербина О. А. О несериальной модификации локального алгоритма декомпозиции задач дискретной оптимизации // Динамические системы. – 2005. – Вып. 19. – С. 179-190.

5. BOOST, minimum degree ordering algorithm. – URL: http://www.boost.org

6. George J.A. Nested dissection of a regular finite element mesh // SIAM J. Numer. Anal. – 1973. – V. 10. – P. 345-367.

7. Karypis G., Kumar V. MeTiS – a software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. Version 4. – University of Minnesota, 1998.

8. Tarjan R. E.,Yannakakis M. Simple linear-time algorithms to test chordality of graphs, test acyclity of hypergraphs, and selectively reduce acyclic hypergraphs // SIAM J. Comput. – 1984. – V. 13. – P. 566-579.

9. CHOMPACK, maximum cardinality ordering algorithm. – URL: http://abel.ee.ucla.edu/chompack/

10. Jégou P., Ndiaye S. N., Terrioux C. Computing and exploiting tree-decompositions for (Max-)CSP // Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming (CP-2005). – 2005. – P. 777-781.

11. Rose D., Tarjan R., Lueker G. Algorithmic aspects of vertex elimination on graphs // SIAM J. on Computing. – 1976. – V. 5. – P. 266-283.

12. Musliu N., Samer M., Ganzow T., Gottlob G. A csp hypergraph library, Technical Report, DBAI-TR-2005-50. – Technische Universität Wien, 2005.

# ДОСТАТОЧНОЕ УСЛОВИЕ ЭФФЕКТИВНОЙ РАЗРЕШИМОСТИ ЗАДАЧИ OPEN SHOP В ТЕРМИНАХ СУММАРНОЙ НАГРУЗКИ

**Севастьянов С.В. (ИМ СО РАН, Новосибирск)**
*seva@math.nsc.ru*
**Черных И.Д. (ИМ СО РАН, Новосибирск)**
*idchern@math.nsc.ru*

Классическая задача теории расписаний open shop формулируется следующим образом. Множество работ $\mathcal{J} = \{J_1, \ldots, J_n\}$ должно быть выполнено на машинах из $\mathcal{M} = \{M_1, \ldots, M_m\}$. Выполнение операции $O_{ji}$ работы $J_j$ машиной $M_i$ занимает время $p_{ji}$. Никакие две операции одной работы или на одной машине не могут выполняться одновременно. Требуется построить допустимое расписание минимальной длины, которая определяется как время завершения наиболее поздней операции ($C_{\max}$). Задача полиномиально разрешима в случае $m = 2$ и является NP-трудной при $m \geq 3$ [1]. При нефиксированном числе машин (когда $m$ является частью входа) задача NP-трудна в сильном смысле [2].

Обозначим множество входов $m$-машинной задачи open shop через $\mathcal{I}_m$. Для каждого $I \in \mathcal{I}_m$ будем использовать следующие обозначения:

- $\ell_i(I) \doteq \sum_j p_{ji}$ — *нагрузка* машины $M_i$;
- $d_j(I) \doteq \sum_i p_{ji}$ — *длина* работы $J_j$;
- $\ell_{\max}(I) \doteq \max_i \ell_i(I)$, $d_{\max}(I) \doteq \max_j d_j(I)$ — наибольшая машинная нагрузка и наибольшая длина работы, соответственно;
- $\Delta(I) \doteq \sum_{i,j} p_{ji} = \sum_i \ell_i(I) = \sum_j d_j(I)$ — *суммарная нагрузка*;
- $\bar{C}(I) \doteq \max\{\ell_{\max}(I), d_{\max}(I)\}$ — нижняя оценка оптимума для входа $I$;
- через $\mathcal{J}(I)$ и $\mathcal{M}(I)$ будем обозначать множества работ и машин во входе $I$.

Обозначение входа $I$ будем опускать в случаях, когда это не вносит разночтений.

Для входа $I$ допустимое расписание длины $\bar{C}(I)$ будем называть *нормальным*; также *нормальным* будем называть вход $I$, для которого существует нормальное расписание. Подмножество входов $\mathcal{K}$, состоящее только из нормальных входов, будем называть *нормальным классом*. Если для некоторого нормального класса $\mathcal{K}$ существует полиномиальный алгоритм построения нормального расписания для любого из его входов, то такой класс будем называть *эффективно-нормальным*.

В статье [3] рассматриваются достаточные условия нормальности входов, описываемые в терминах неравномерности нагрузок машин, и описываются соответствующие (достаточно широкие) эффективно-нормальные классы. Целью данной работы является выявление эффективно-нормальных классов, описываемых в терминах ограничений на суммарную нагрузку $\Delta$.

Рассмотрим класс входов $\mathcal{K}_m(\alpha) \doteq \{I \in \mathcal{I}_m | \Delta(I) \leq \alpha \bar{C}(I)\}$. Для каждого $m \geq 3$ исследуем, при каком наибольшем значении $\alpha$ класс $\mathcal{K}_m(\alpha)$ является нормальным, т.е. найдем наиболее широкие нормальные классы вида $\mathcal{K}_m(\alpha)$. Заметим, что для случая $m = 2$ такая постановка вопроса неинтересна, поскольку множество всех входов $\mathcal{I}_2$ является эффективно-нормальным классом [1].

В дальнейшем будем пользоваться следующими преобразованиями входов.

**Определение 1.** Будем говорить, что вход $I'$ получен из входа $I$ с помощью *склеивания множества машин* $\bar{M}$, если $\mathcal{J}(I') = \mathcal{J}(I)$ и $\mathcal{M}(I') = \mathcal{M}(I) \setminus \bar{M} \cup \{M_{i'}\}$, где $p_{ji'} = \sum_{M_i \in \bar{M}} p_{ji}$.

Заметим, что допустимое расписание для "склеенного" входа $I'$ можно интерпретировать как допустимое расписание такой же длины для исходного входа $I$. Справедлива следующая

**Лемма 1.** *Для любого $m \geq 3$ и $\alpha > 2$ класс $\mathcal{K}_m(\alpha)$ не является нормальным.*

**Доказательство.** Рассмотрим следующий вход с четырьмя работами. Работа $J_1$ имеет три операции на машинах $M_1$, $M_2$ и $M_3$ длительности $1/3$, операции на остальных машинах — нулевые. Каждая из работ $J_k$, $k = 2, 3, 4$, имеет единственную ненулевую операцию длительности $1/3 + \varepsilon$ на машине $M_{k-1}$. Для этого входа $\bar{C} = 1$, $\Delta = 2 + 3\varepsilon$. Докажем, что он не является нормальным. Предположим обратное, что для этого входа существует расписание длины $\bar{C}$. Поскольку $d_1 = \bar{C}$, операции работы $J_1$ выполняются в этом расписа-

нии одна за другой, без промежуточных ожиданий. При этом одна из них выполняется (на некоторой машине $M_k$) в интервале $[1/3, 2/3]$. Ясно однако, что ненулевую операцию работы $J_{k+1}$ невозможно выполнить на этой же машине ни в одном из оставшихся свободных интервалов $[0, 1/3], [2/3, 1]$, что означает невозможность выполнения совокупности работ в интервале $[0, \bar{C}]$. Полученное противоречие доказывает лемму. ■

Рассмотрим объединение классов $\mathcal{K}(\alpha) \doteq \bigcup_m \mathcal{K}_m(\alpha)$. Справедлива следующая

**Теорема 2.** *$\mathcal{K}(2)$ является эффективно-нормальным классом.*

**Доказательство.** Рассмотрим вход $I \in \mathcal{K}(2)$. В силу симметричности понятий машина/работа в системе open shop, без ограничения общности можем считать, что $\bar{C}(I) = \ell_1(I)$. Поскольку $\sum \ell_i(I) \leq 2\bar{C}(I)$, справедливо $\sum_{i=2}^{m} \ell_i(I) \leq \bar{C}(I)$. Проведем склеивание множества машин $\bar{M} = \{M_2, \ldots, M_m\}$. Поскольку $\ell_1(I') = \ell_1(I) = \bar{C}(I)$, $\ell_2(I') \leq \bar{C}(I)$, а длины работ нового входа $I'$ совпадают с длинами соответствующих работ из $I$, имеем $\bar{C}(I') = \bar{C}(I)$.

Поскольку $I' \in \mathcal{I}_2$, оптимальное расписание длины $C(I')$ может быть построено алгоритмом Гонзалеза-Сани [1] за время, линейное от числа работ. Это расписание можно интерпретировать как допустимое расписание длины $C(I)$ для исходного входа $I$. ■

Заметим, что в силу леммы 1 и теоремы 2 любой нормальный класс вида $\mathcal{K}_m(\alpha)$ является подмножеством $\mathcal{K}(2)$, т.е. для любого $m \geq 3$ наиболее широким нормальным классом вида $\mathcal{K}_m(\alpha)$ является $\mathcal{K}_m(2)$.

Работа выполнена при финансовой поддержке РФФИ, грант №08-01-00370.

**Литература**

1. Gonzalez, T., Sahni, S. Open shop scheduling to minimize finish time // J. Assoc. Comp. Math. 23, 1976, p. 665–679.

2. Williamson, D.P. et al. Short shop schedules // Oper. Res. 1997, V. 45, N. 2, p. 288-294.

3. Kononov, A., Sevastianov, S., Tchernykh, I. When difference in machine loads leads to efficient scheduling in open shops // Annals of Oper. Res. 92, 1999, p. 211–239.

# ПАРАЛЛЕЛЬНОЕ РАЗЛОЖЕНИЕ ХОЛЕЦКОГО РАЗРЕЖЕННОЙ МАТРИЦЫ

**Старостин Н.В., Сафонова Я.Ю. (Нижегородский государственный университет имени Н.И. Лобачевского)**
*nvstar@mail.ru, safonova.yana@gmail.com*

## 1. Введение

Системы линейных уравнений $Ax = b$ с симметричной положительно определенной матрицей $A$ возникают при численном решении задач математической физики. Для решения систем такого рода применяют разложение Холецкого вида $A = U^T U$, где $U$ — верхнетреугольная матрица. Тогда решение исходной системы эквивалентно решению двух систем: $U^T y = b, Ux = y$. Классический алгоритм нахождения разложения, предложенный в [1], последовательно формирует элементы новой строки, начиная с первой, и использует при этом значения в найденных строках. Такой подход не обладает параллелизмом, так как независимый расчет строк приведет к некорректному результату. Однако, если матрица $A$ является разреженной, то некоторые строки могут не зависеть от значений предыдущих строк, и выделение независимых друг от друга групп позволяет организовать параллельный расчет. В [2] дано определение дерева исключения матрицы, как отражения ее структуры разреженности, определяющей зависимость между строками, и сформулированы теоремы, позволяющие сконструировать параллельный алгоритм нахождения разложения по известному дереву исключения. Задача построения дерева исключения совпадает по сложности с задачей определения портрета матрицы $U$ [3], при этом вид полученного дерева может не допускать эффективного распараллеливания. В этом случае предлагается предварительно переупорядочивать матрицу $A$ с помощью алгоритмов [1], в ходе выполнения которых дерево исключения матрицы становится известным и хорошо сбалансированным. К таким алгоритмам относят методы семейства вложенных сечений. В данной статье рассматривается параллельная схема общего вида для решения поставленной задачи, позволяющая достичь ускорения $log_2 p$, где $p$ — число независимых вычислителей.

## 2. Связь дерева исключения с переупорядочиванием матрицы

Пусть $G(A) = (V, E)$ — граф матрицы $A$. Деревом исключения называется дерево, чье множество вершин совпадает с $V$, а ребро

$(v_i, v_j)$ существует тогда и только тогда, когда $i = min\{k : u_{jk} \neq 0 \ \& \ k > j\}$. Таким образом, $v_i$ является родителем $v_j$, если первый недиагональный элемент строки $j$ располагается в столбце $i$. Такое определение дает понятие зависимости между строками. Если вершина $v_i$ является предком вершины $v_j$, то значения в строке $i$ должны быть вычислены после значений в строке $j$, так как зависят от них. Ресурсом для получения ускорения является параллельное вычисление строк, соотвествующие вершины которых не являются по отношению к друг другу предком или потомком. При этом, чем меньше высота дерева исключения при заданном количестве вершин, тем более высокое ускорение можно получить. Переупорядочивание матрицы методом вложенных сечений дает бинарное дерево исключения [4], причем настройка параметров алгоритма позволяет получать сбалансированность между поддеревьями одного уровня. Другим критерием при выборе алгоритма переупорядочивания является заполнение, возникающее в матрице $U$, так как при решении большеразмерных систем сохранение разреженности более важная задача, чем распараллеливание. В статье приведены результаты, демонстрирующие, что для типовых задач, возникающих на практике, метод вложенных сечений дает не худшее заполнение, чем другие известные алгоритмы [1].

**3. Программная реализация**

На основе приведенных рассуждений предлагается программная реализация, основанная на предварительном переупорядочивании исходной матрицы методом вложенных сечений. Итогом переупорядочивания является матрица перестановок и дерево исключения. В силу эквивалентости задач определения дерева исключения и символической факторизации будем считать, что портрет матрицы $U$ является известным. Для дальнейшего параллельного выполнения необходимо сделать разбиение дерева исключения. Один из способов разбиения — выделение поддеревьев одинаковой высоты. Высота поддеревьев — эвристический параметр, влияющий на балансировку между потоками, и его оптимальное значение зависит от конфигурации вычислительной системы. Для матриц с порядком от 50000 до 1000000 рекомендуется выбирать его в диапазоне от 50 до 100. Между поддеревьями необходимо установить отношение предок–потомок, и каждое поддерево связать с задачей. Под задачей понимается атомарный набор операций, который будет последо-

вательно выполняться на одном вычислителе для всех вершин поддерева, ей соотвествующего.
Пример задачи:

1. Для всех потомков текущего поддерева запустить параллельное выполнение соотвествующих задач.

2. Синхронизировать окончание выполнения порожденных задач.

3. Для каждой вершины поддерева выполнить шаги вычислительной фазы.

Запуск параллельного алгоритма инициализирует задача, соотвествующая корневому поддереву. Данная схема подходит для численой факторизации и решения системы $U^T y = b$ при обходе вершин от листьев поддерева к корню. Решение системы $Ux = y$ соотвествует схеме с последовательностью шагов 3, 1, 2 и обратным порядком обхода вершин в поддереве для шага 3.

**4. Результаты вычислительных экспериментов**

В первой части экспериментов сравнивались показатели заполнения для разных алгоритмов переупорядочивания. Результаты показывают, что метод вложенных сечений дает сравнимое с другими алгоритмами заполнение.

| Название | CM | RCM | King | Sloan | ND |
|---|---|---|---|---|---|
| shallow_water2 | 0,00656 | 0,00656 | 0,00739 | 0,00654 | 0,00063 |
| parabolic_fem | 0,00163 | 0,00163 | 0,00163 | 0,00163 | 0,00019 |
| tmt_sym | 0,00242 | 0,00236 | 0,00266 | 0,00225 | 0,00018 |
| thermomech_dM | 0,00181 | 0,0017 | 0,00178 | 0,00512 | 0,00045 |

где CM — прямой алгоритм Катхилл–Макки, RCM — обратный алгоритм Катхилл–Макки, King — алгоритм Кинга, Sloan — алгоритм Слоуна, ND — метод вложенных сечений.
Во второй части экспериментов тестировалась описанная выше схема распараллеливания. Следует отметить, что верхняя оценка для ускорения такой схемы на $p$ вычислителях $log_2 p$. Результаты решения СЛАУ на 8 потоках показывают ускорение, близкое к теоретически возможному:

| Название | $N$ | $S_1$ | $S_2$ | $H$ | $H^*$ |
|---|---|---|---|---|---|
| shallow_water2 | 81920 | 1,65 | 2,62 | 930 | 100 |
| parabolic_fem | 525825 | 1,84 | 2,83 | 2826 | 75 |
| tmt_sym | 726713 | 1,62 | 2,59 | 3932 | 100 |
| thermomech_dM | 204316 | 2,79 | 3,16 | 1293 | 50 |

где $N$ — порядок матрицы, $S_1$ — ускорение численной факторизации, $S_2$ — ускорение при решении систем, $H$ — высота дерева исключения, $H^*$ — высота поддеревьев в разбиении.

**Литература**

1. Писсанецки С. Технология разреженных матриц. – М.: Мир, 1988, с. 76-79, 115, 163.

2. Heggerness P. Minimizing fill-in size and elimination tree height in parallel Cholesky factorization. – Bergen: Department of Informatics University of Bergen, 1992, p. 15-17.

3. van Grondelle J. Symbolic Sparse Cholesky Factorization Using Elimination Trees. – Utrecht: Department of Mathematics Utrecht University, 1999, p. 3-6.

4. Сафонова Я. Ю. Использование графовых моделей при распараллеливании метода Холецкого при решении разреженных симметричных СЛАУ. // Материалы XVI международной конференции Проблемы теоретической кибернетики. – Нижний Новгород: Изд-во ННГУ, 2011, с. 426-430.

# ФОРМАЛИЗАЦИЯ ЗАДАЧИ ПОИСКА ОБЪЕКТОВ НА ВЕКТОРНОЙ СЦЕНЕ

**Фофанов В.Б., Жизневский А.Н. (Казанский (Приволжский) Федеральный Университет)**

*Viatcheslav.Fofanov@ksu.ru, write2aracon@mail.ru*

**Введение**

Поиску на сцене объектов определенного вида по ее изображению посвящено довольно большое количество публикаций. Предлагаемые в них методы являются, как правило, эвристическими. В настоящей работе излагаются формализация и решение этой задачи в рамках теоретико-вероятностного подхода. При этом предполагается, что объекты являются пятнами, а исход-ной информацией о сцене служит набор ее изображений. Поиск объектов предлагается проводить в три этапа. На первом определяются квадратные фрагменты сцены, названные зонами интереса. Каждая зона содержит один объект и его окружение (фон). На втором этапе (сегментации) проводится классификация пикселей зоны интереса на два класса. Пиксели, оказавшиеся в классе с именем объект, используются на третьем этапе для вычисления геометрических признаков объекта и принятия окончательного решения о его присутствии на сцене.

**1. Модель сцены**

Пусть $(\Omega, A, P)$ — вероятностное пространство, $Y = \{0, 1, \ldots, n-1\}$ — множество из $|Y| = n$ объектов и $A$ — конечное подмножество на целочисленной решетке $Z^2$. Назовем объектом сцены с проекцией $A$ семейство $\boldsymbol{\xi}_A = (\boldsymbol{\xi}_a)_{a\in A}$ векторных случайных величин $\boldsymbol{\xi}_a = (\xi_a^j)_{1\le j\le \nu}$, определенных на $(\Omega, A, P)$ и принимающих значения в $\mathbf{Y} = Y^\nu$. Изображением объекта $\xi_A$ будет называться семейство $\mathbf{x}_a = (\mathbf{x}_a)_{a\in A}$ из $\mathbf{Y}^A$. Объект считается заданным, если на множестве $\mathbf{Y}^A$ его изображений задано распределение вероятностей $P_{\mathbf{Y}^A} = (p_{\mathbf{Y}^A}(\mathbf{x}_A))_{\mathbf{x}_A\in\mathbf{Y}^A}$. Совокупность объектов назовем векторной сценой, если их проекции попарно не пересекаются, а их сумма равна $Z^2$. Таким об-разом, сцена является векторным случайным полем $(\boldsymbol{\xi}_z)_{z\in Z^2}$. В частном случае, когда случайные величины являются скалярными, сцена также будет называться скалярной. В качестве изображения $\mathbf{x}_z = (\mathbf{x}_z)_{z\in Z^2}$ сцены естественно рассматривать реализации (выборочные поверхности) случайного поля. Существование указанных сцен вытекает из следующей теоремы.

**Теорема 1.** *Пусть на $Z^2$ задано разбиение, состоящее из конечных попарно непересекающихся подмножеств, называемых проекциями, и пусть каждой проекции $A \subset Z^2$ поставлено в соответствие распределение вероятностей $P_{\mathbf{Y}^A}$ на множестве $\mathbf{Y}^A$. Тогда существует вероятностное пространство $(\Omega, A, P)$ и векторная сцена $(\boldsymbol{\xi}_z)_{z \in Z^2}$ такая, что*

$$P\{\omega \in \Omega : \boldsymbol{\xi}_A(\omega) = \mathbf{x}_A\} = p_{\mathbf{Y}^A}(\mathbf{x}_A)$$

*для любой проекции $A$ и любого $\mathbf{x}_A \in \mathbf{Y}^A$. Кроме того, если $A$ и $B$ — проекции разных объектов, то для любых $a \in A$ и $b \in B$ случайные величины $\boldsymbol{\xi}_a$ и $\boldsymbol{\xi}_b$ - независимы.*

Далее предполагается, что каждый объект $\boldsymbol{\xi}_A$ является фрагментом однородного случайного поля с вектором средних значений $\mathbf{m}_A \in \mathbf{Y}$, для которого выполняются условия эргодической теоремы Слуцкого. Это позволяет использовать среднее арифметическое значение $\bar{\mathbf{x}}_A = \frac{1}{|A|} \sum_{a \in A} \mathbf{x}_a$, вычисленное по изображению объекта $\mathbf{x}_A$ в качестве статистической оценки для $\mathbf{m}_A$ .

**2. Поиск зон интереса**

Пусть $d$ — евклидово расстояние на $Z^2$. Точку $a$ из $A$ назовем граничной, если $d(a, A^C) = 1$. Совокупность $Fr(A)$ граничных точек $A$ будет называться его границей. Объект $\boldsymbol{\xi}_A$, проекция которого является односвязным множеством, назовем пятном, если существует квадрат $Q$ на $Z^2$ такой, что $A \subset (Q \setminus Fr(Q))$, если $E\xi_z = \mathbf{m}_{Q \setminus A}$, $z \in Q \setminus A$, и если $d(\mathbf{m}_A, \mathbf{m}_{Q \setminus A}) > 0$. Семейство $\boldsymbol{\xi}_{Q \setminus A}$ будет называться окрестность пятна, а $\boldsymbol{\xi}_Q$ — его зоной интереса.

Пусть $A$ — проекция пятна $\boldsymbol{\xi}_A$, $d(A)$ — его диаметр, а $B(a, r)$ — окрестность (круг с центром $a$ и радиусом $r$), принадлежащая $A$. Легко видеть, что граница квадрата $Q \subset Z^2$ с центром $a$ и стороной $l$, $l \geq 2(d(A) - r + 1)$, не имеет общих точек с $A$. Если $\boldsymbol{\xi}_Q$ — зона интереса для $\boldsymbol{\xi}_A$, то изображение $\mathbf{x}_{Fr}$ границы можно использовать для получения оценок среднего $\mathbf{m}_{Q \setminus A}$ фона. В самом деле, пусть $n = |B(a, r)|$, разделим границу $Fr(Q)$ на $s$ непересекающихся связных частей $Fr_j$, $1 \leq j \leq s$, по $n$ точек и вычислим $s$ средних арифметических значений $\bar{\mathbf{x}}_j = \frac{1}{n} \sum_{z \in Fr_j} \mathbf{x}_z$, $1 \leq j \leq s$. С другой стороны, среднее арифметическое значение $\bar{\mathbf{x}}_a = \frac{1}{n} \sum_{z \in B(a,r)} \mathbf{x}_z$ будет оценкой среднего $\mathbf{m}_A$ пятна. Сопоставим каждому квадрату $Q$ со

стороной $l$ и центром $a \in Z^2$ признак $f_{\{a\}}$, определенный равенством

$$f_{\{a\}}(\mathbf{x}) = \sum_{j=1}^{s} I_{]0,+\infty[}(d(\mathbf{x}_a, \mathbf{x}_{Fr}) - d(\mathbf{x}_j, \mathbf{x}_{Fr})).$$

Если $\Theta_1$ — множество зон интереса, а $\Theta_2$ — квадраты из пикселей фона, то для сцен, полученных скользящим суммированием по окрестности с радиусом $\hat{r}$, имеет место следующий результат.

**Теорема 2.** *Пусть $P(\Theta_1)$ и $P(\Theta_2)$ — априорные вероятности, $Fr_j$, $1 \le j \le s$, — связные фрагменты границы $Fr(Q)$, содержащие по $n = |B(a,r)|$ пикселей, такие, что $d(Fr_i, Fr_j) > 2\hat{r}$, $f_{\{a\}}$ — признак квадрата $\boldsymbol{\xi}_Q$, соответствующий $r$ и $s$. Тогда семейство $h_n^s : Y_f \to \{1,2\}$ решающих правил вида*

$$h_n^s(f_{\{a\}}(\mathbf{x})) = \begin{cases} 1, & f_{\{a\}}(\mathbf{x}) = s \\ 2, & f_{\{a\}}(\mathbf{x}) < s \end{cases}$$

*классификации квадратов из $\Theta_1 + \Theta_2$ стремится при $r \to +\infty$ и $s \to +\infty$ к байесовскому решающему правилу с вероятностью ошибки, равной нулю.*

**3. Сегментация**

Пусть $\boldsymbol{\xi}_Q$ — зона интереса векторной сцены. Пусть $Fr_j$, $1 \le j \le s$, — семейство связных попарно не пересекающихся фрагментов из точек на $Fr(Q)$. По теореме Слуцкого, средние арифметические значения

$$\bar{\mathbf{x}}_{Fr} = \frac{1}{|Fr(Q)|} \sum_{t \in Fr(Q)} \mathbf{x}_t \text{ и } \bar{\mathbf{x}}_j = \frac{1}{Fr_j} \sum_{t \in Fr_j} \mathbf{x}_t,\ 1 \le j \le s,$$

являются оценками координат вектора $\mathbf{m}_{Q \setminus A}$. Пусть $B(z,r)$ — окрестность из $Q \setminus Fr(Q)$, а $\bar{\mathbf{x}}_z = \frac{1}{|B(z,r)|} \sum_{t \in B(z,r)} \mathbf{x}_t$ — среднее арифметическое значение. Если $f_{\{z\}}(\mathbf{x}) = s$, то пиксель $\boldsymbol{\xi}_z$ будем относить к объекту $\boldsymbol{\xi}_A$, а при $f_{\{z\}}(\mathbf{x}) < s$ — к фону. Присвоим номер 1 подмножеству пикселей, образующих объект, и номер 2 - подмножеству всех остальных пикселей зоны интереса. Тогда изложенный способ сегментации можно рассматривать как классификацию ее пикселей на два класса с использованием решающего правила

$$h_r^s(f_{\{z\}}(\mathbf{x})) = \begin{cases} 1, & f_{\{z\}}(\mathbf{x}) = s \\ 2, & f_{\{z\}}(\mathbf{x}) < s \end{cases}$$

Пусть $A^\circ$ и $(Q\setminus A\setminus Fr(Q))^\circ$ — внутренние пиксели объекта и фона соответственно. Для сцен, полученных скользящим суммированием по окрестности с радиусом $\hat{r}$, имеет место следующий результат.

**Теорема 3.** *Пусть $\boldsymbol{\xi}_Q$ — зона интереса, $P(A^\circ)$ и $P((Q \setminus A \setminus Fr(Q))^\circ)$ — априорные вероятности классов, $Fr_j$, $1 \le j \le s$, — связные фрагменты границы $Fr(Q)$, состоящие из $n = |B(a,r)|$ пикселей каждый, и такие, что $d(Fr_i, Fr_j) > 2\hat{r}$, $f_{\{z\}}$ — признак, соответствующий $r$ и $s$. Тогда семейство $h_r^s$ решающих правил стремится при $r \to +\infty$ и $s \to +\infty$ к байесовскому решающему правилу $h$ с вероятностью ошибки $e(h) = 0$.*

# МАТЕМАТИЧЕСКИЕ МОДЕЛИ ЭКОНОМИЧЕСКИХ ПРОЦЕССОВ

**Черемных Ю.Н.**

Содержательные области и проблематика экономико-математического моделирования исторически формировались постепенно: проблемы ценообразования в различных рыночных структурах (от чистой конкуренции до чистой монополии), рационального поведения потребителя на рынке, максимизации прибыли, спроса и предложения, относящиеся традиционно к микроэкономике. К макроэкономике относятся вопросы, связанные со статикой и динамикой инвестиций, процентной ставки, безработицы и валового национального продукта. Отдельно следует отметить проблему рыночного равновесия (на микро- и макроэкономическом уровнях). Сюда необходимо добавить вопросы структуризации данных и их преобразования в модельную информацию, анализа влияния отдельных факторов на поведения ключевых экономических показателей (предмет математической статистики и эконометрики), совокупность задач, связанных с операциями на финансовых рынках (т.е. на рынках ценных бумаг и финансовых услуг).

Применяемые математические методы и средства для решения разнообразных экономических задач отличаются большим разнообразием, что связано с широтой круга содержательных экономических областей. В дополнительных комментариях не нуждается то обстоятельство, что экономическая материя много сложнее различных областей естественно-научной материи.

Теория систем линейных алгебраических уравнений — основной инструмент анализа статической межотраслевой модели, предложенной в 1930-е гг. американским экономистом В.В. Леонтьевым (Премия по экономике имени Нобеля, 1973). Статическая межотраслевая модель (модель затраты —выпуск) представляет собой развитие идеи межотраслевого описания национальной экономики Советской России, реализованной группой советских экономистов в середине 1920-х гг.

Метод неопределенных множителей, опубликованный французским математиком Ж. Л. Лагранжем на рубеже 18–19 вв., активно использовался и используется экономистами для решения разнообразных задач — в основном микроэкономики: задачи рационального

поведения потребителя на рынке, задача рационального распределения ограниченных ресурсов и задача издержек при фиксированном объеме выпускаемой продукции. В этих задачах множитель Ж. Л. Лагранжа имеет глубокий экономический смысл. В теории потребления он связывает величину дохода потребителя с уровнем полезности, в теории производства — лимит на ресурсы с объемом выпускаемой продукции.

Линейное программирование возникло под давлением экономической проблематики (проблема рационального использования ограниченных ресурсов) и было отмечено Премией по экономике имени Нобеля, которую получили советский математик Л. В. Канторович и американский математик Т. Купманс. На основе симплексного метода, предложенного американским математиком Дж. Данцигом, и на базе использования ЭВМ было решено большое число оптимизационных задач с экономическим и военным содержанием.

Линейное программирование долгое время представляло собой интенсивно развивающуюся область прикладной математики (нелинейное программирование).

Теория обыкновенных дифференцированных и разностных уравнений широко востребована экономистами для анализа разнообразных эволюционных процессов, в частности, динамики рыночных цен и ее качественного анализа с использованием теории устойчивости А. М. Ляпунова.

Положения теории оптимального управления находят активное применение в решении и анализе многих задач макроэкономики.

Для решения широких классов разнообразных экономических задач интенсивно применяются разделы теории вероятностей и математической статистики.

Следует отметить, что примерно до середины 20 в. уровень применения математики в экономических исследованиях заметно отставал от аналогичного уровня применения математики в естественнонаучных областях. После публикации в 1954 г. журналом “Эконометрика” статьи “Существование равновесия в конкурентной экономике” американского экономиста К. Эрроу (лауреат Премии по экономике имени Нобеля (1972) и французского математика Ж. Дебре (лауреат Премии по экономике имени Нобеля (1983) ситуация заметно изменилась благодаря высокому математическому уровню статьи К. Эрроу и Ж. Дебре. Уровень был вполне сопоставим с математическим

уровнем самых продвинутых статей естественно-научного профиля. Добавим, что Ж. Дебре был членом группы Н. Бурбаки, работавшей во Франции после окончания Второй мировой войны.

Исторически первой монографией экономико-математического содержания была работа "Исследования математических принципов теории богатства" французского математика А. Курно (1838). А. Курно предложил модель олигополии и равновесие этой модели, которое позже получило имя равновесия А. Курно. В модели А. Курно фирмы принимают решения об объеме выпускаемой продукции в условиях полной неопределенности, ибо не знают, какие решения примут конкуренты. В связи с этим вопрос о статусе оптимального решения этой модели является до сих пор открытым. Равновесие А. Курно — это субоптимальное решение модели, которое обладает свойством, что в одиночку ни одной фирме не выгодно выходить из этого равновесия.

Позднее, в 1950 г., американский математик Дж. Нэш (лауреат Премии по экономике имени Нобеля, 1994) предложил понятие равновесия некооперативной игры конечного числа игроков, которое позже получило его имя и которое обобщает равновесие А. Курно.

Во второй половине 19 в. французский экономист Л. Вальрас предложил понятие конкурентного равновесия в многопродуктовой экономике, состоящей из двух сфер: сферы производства и сферы потребления. Сфера производства состоит из конечного числа фирм, каждая из которых максимизирует свою прибыль. Сфера потребления состоит из конечного числа потребителей, каждый из которых максимизирует свою функцию полезности. Суть конкуренции по Л. Вальрасу в том, что фирмы и потребители каждый сам по себе решают свою задачу максимизации. Вопрос формулировался так: существуют ли такие цены (называемые ценами равновесия), ориентируясь на которые каждая фирма и каждый потребитель решают задачу максимизации и при этом не будет дефицита ни по одному продукту. Полное положительное решение этой задачи К. Эрроу и Ж. Дебре (как уже отмечалось выше) опубликовали в 1954 г. Они использовали теорему о неподвижной точке точечно- множественного отображения японского математика Ш. Какутани, опубликованную в 1941г .

В конце 19 в. итальянский экономист В. Парето предложил фундаментальное понятие, называемое сейчас Парето- эффектив-

ностью (Парето-оптимальностью), суть которого в следующем: совокупность потребительских наборов Парето-эффективна, если не существует другой совокупности потребительских наборов, на которых значения функции полезности не меньше, чем на первоначальных наборах и, по крайней мере, одно из этих неравенств должно быть строгим.

В 1945 г. на английском языке был опубликован перевод с немецкого статьи "Модель общего экономического равновесия" американского математика Дж. фон Неймана, в которой было предложено понятие динамического равновесия общей модели производства в матричной форме, охватывающей производственную и монетарную сферы. Статья Дж. фон Неймана оказала большое влияние на развитие математической экономики после окончания Второй мировой войны.

В 1958 г. в книге "Линейное программирование и экономический анализ" трех американских авторов (Дорфман Р., Сэмуэльсон П. (лауреат Премии по экономике имени Нобеля, 1970), Солоу Р. (лауреат Премии по экономике имени Нобеля, 1987) было описано магистральное свойство оптимальных траекторий валовых выпусков и цен, суть которого состояла в том, что в случае продолжительного временного промежутка модели ее оптимальная траектория выпусков ( для оптимальной траектории цен ситуация аналогична) состоит из трех участков. Первый участок — это отрезок оптимальной траектории, который приближается к траектории максимального постоянного пропорционального роста, расположенной на луче, называемом магистралью; второй участок — это отрезок оптимальной траектории, который близок (в смысле углового расстояния) к траектории максимального постоянного пропорционального роста; третий участок — это отрезок оптимальной траектории, который может отойти от траектории максимального постоянного пропорционального роста. Основные характеристики траектории максимального постоянного пропорционального роста (темп роста и структура) определяются технологическим множеством модели, т. е. эндогенно. Теоретическое значение магистрального свойства состоит в том, что в случае продолжительного временного горизонта модели оптимум достигается через максимальный постоянный пропорциональный рост, практическое значение магистрального свойства в том, что оно позволяет дать рациональное решение "проблемы хвоста" и проблемы

сглаживания оптимальной траектории.

В заключение подчеркнем наличие взаимного влияния экономической проблематики и отдельных разделов математики: математика использовалась и используется для решения задач экономической теории и хозяйственной практики, а экономические задачи выступали в качестве генераторов новых областей как прикладной (линейное программирование), так и чистой математики (выпуклый анализ).

Бурное развитие компьютеризации привело к повышению эффективности применения математики в решении прикладных экономических задач.

**Литература**

1. Экономико математический энциклопедический словарь. — М.: Большая Российская энциклопедия, Издательский Дом “ИНФРА-М”, 2003.

# ДОСТАТОЧНОЕ УСЛОВИЕ РАЗРЕШИМОСТИ ДВУХМАШИННОЙ ЗАДАЧИ OPEN SHOP С МАРШРУТИЗАЦИЕЙ И РАЗРЕШЕНИЕМ ПРЕРЫВАНИЙ

**Черных И.Д. (ИМ СОРАН, Новосибирск)**
*idchern@math.nsc.ru*
**Кузеванов М.А. (НГУ, Новосибирск)**
*theredstar@mail.ru*

Рассматриваемая задача обобщает две классические задачи дискретной оптимизации: метрическую задачу комивояжера и задачу Open Shop с прерываниями. Имеется $n$ работ $J_1, \ldots, J_n$, каждая из которых должна быть обработана каждой из $m$ машин $M_1, \ldots, M_m$ в произвольном порядке. Операция $O_{ji}$ обработки работы $J_j$ машиной $M_i$ занимает $p_{ji}$ единиц времени. Операцию можно прерывать в любой момент времени и возобновить ее выполнение позднее. Работы расположены в вершинах транспортной сети, описываемой реберно-взвешенным графом $G = \langle V, E \rangle$, расстояние между вершинами $v_i$ и $v_j$ обозначается через $\tau_{ij}$. Множество работ, расположенных в вершине $v_k$, обозначим через $\mathcal{J}_k$. Одна из вершин является *базой*; изначально все машины находятся в базе и должны вернуться туда после выполнения всех своих операций. Машины передвигаются с единичной скоростью. Требуется составить допустимое расписание, минимизирующее время возвращения всех машин на базу после выполнения всех операций. Задача рассматривается в двух постановках: с заданной базой или с выбираемой базой, обозначаемых соответственно $ROm|pmtn|F_{\max}$ и $ROm|pmtn, free - depot|F_{\max}$.

Задача является NP-трудной в сильном смысле даже в случае $m = 1$ (в этом случае она эквивалентна метрической задаче комивояжера). Двухмашинная задача является полиномиально разрешимой в случае, когда транспортная сеть состоит из двух вершин [1]. Алгоритмическая сложность двухмашинной задачи на треугольнике является открытым вопросом. Двухвершинная задача с нефиксированным числом машин является NP-трудной в сильном смысле [1]. Целью данной работы является описание полиномиально-разрешимых подклассов задач $RO2|pmtn|F_{\max}$ и $RO2|pmtn, free - depot|F_{\max}$.

Поскольку задача включает в себя задачу комивояжера, во всех ее полиномиально разрешимых подслучаях граф $G$ и/или функция расстояний $\tau$ должны позволять решить задачу комивояжера за полиномиальное время. Например, число вершин графа может быть

фиксированным, или структура графа позволяет упростить поиск кратчайшего обхода.

Введем следующие обозначения. Через $\ell_i \doteq \sum\limits_j p_{ji}$ обозначим *нагрузку* машины $M_i$ через $d_j \doteq \sum\limits_i p_{ji}$ обозначим *длину* работы $J_j$. Длину кратчайшего обхода графа $G$ обозначим через $T^*$. Величина $\tilde{F} \doteq \max\{T^* + \max \ell_i, \max d_j\}$ является нижней оценкой оптимума задачи в обоих постановках.

**Задача с выбираемой базой.** Справедлива следующая

**Теорема 1.** *Пусть кратчайший обход графа $G$ может быть найден за время $t_{TSP}$. Пусть суммарная длина работ в одной из вершин не меньше $\tilde{F}$. Тогда для задачи $RO2|pmtn, free-depot|F_{\max}$ оптимальное расписание имеет длину $\tilde{F}$ и такое расписание с не более чем одним прерыванием может быть найдено за время $O(n+t_{TSP})$.*

Опишем идею алгоритма. В качестве базы выберем вершину с наибольшей суммарной длиной работ. Найдем кратчайший обход $R$ графа $G$ и переобозначим вершины в порядке обхода $R$ начиная с базы. Заметим, что поскольку $\sum\limits_j d_j = \ell_1 + \ell_2 \le 2(\tilde{F} - T^*)$, справедливо

$$\sum_{J_j \notin \mathcal{J}_1} d_j \le \tilde{F} - 2T^*. \tag{1}$$

Проведем процедуру "огрубления" исходного примера следующим образом. Для каждой вершины $v_k$, $k \neq 1$, заменим все работы из $\mathcal{J}_k$ одной новой работой $J'_k$, операции которой имеют длительность $p'_{ki} = \sum\limits_{J_j \in \mathcal{J}_k} p_{ji}$. В вершине $v_1$ проведем "склеивание" работ следующим образом. Пусть $\mathcal{J}_1 = \{J_1, \ldots, J_r\}$. Если $\sum\limits_{J_j \in \mathcal{J}_1} = \tilde{F}$, заменим все работы из базы на одну так же, как в остальных вершинах. В противном случае выберем наибольшее $l$ такое, что $\sum\limits_{j=1}^{l} d_j \le \tilde{F}$. Заменим работы $J_1, \ldots, J_l$ на одну работу $J'_{1-1}$, переобозначим работу $J_{l+1}$ через $J'_{1-2}$, и заменим работы $J_{l+2}, \ldots, J_r$ на новую работу $J'_{1-3}$ (если $l+1 < r$). Заметим, что в новом примере длины всех работ не превосходят $\tilde{F}$. Без ограничения общности считаем, что в новом примере

в базе ровно три работы (при необходимости добавим фиктивные работы с операциями нулевой длительности).

Обозначим операции работы $J'_j$ нового примера через $a_j$, $b_j$. Теми же буквами будем обозначать длительности этих операций. При необходимости, перенумеруем работы из базы и/или машины так, чтобы выполнялось

$$a_{1-2} \geq b_{1-1},\ a_{1-3} \geq b_{1-2}. \tag{2}$$

Построим расписание по следующей схеме. Машина $M_1$ сначала выполняет операции работ $J'_{1-1}$, $J'_{1-2}$ и $J'_{1-3}$, потом следует по маршруту $R$, выполняя операции в каждой вершине, и возвращается на базу. Машина $M_2$ сначала переезжает в вершину $v_2$, выполняет соответствующую операцию и продолжает движение по маршруту $R$, выполняя операции в каждой вершине. После возвращения на базу выполняет операции работ $J'_{1-1}$, $J'_{1-2}$ и $J'_{1-3}$. При этом для работ из базы выполняется сначала операция первой машины, потом второй, для остальных работ — в обратном порядке.

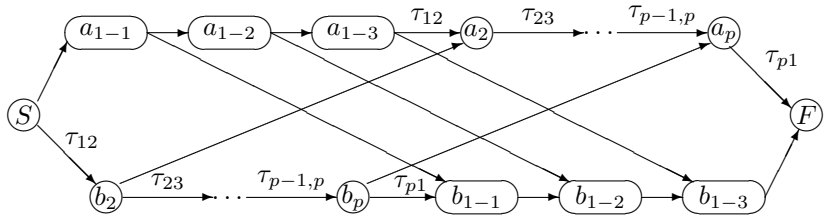



Рис. 1: Схема $N$ построения расписания.

Длина раннего расписания $S$, построенного по схеме $N$, совпадает с длиной некоторого критического пути в $N$ из $S$ в $F$. Заметим, что если критический путь не содержит работ из базы, то в силу (1) его длина не превосходит $\tilde{F}$. Если критический путь содержит только операции одной машины, то его длина по определению $\tilde{F}$ также не превосходит $\tilde{F}$.

Среди оставшихся путей из $S$ в $F$ в силу (2) критическим может быть только $S \to a_{1-1} \to a_{1-2} \to a_{1-3} \to b_{1-3} \to F$. Пусть его длина превышает $\tilde{F}$. Тогда расписание $S$ имеет следующий вид:

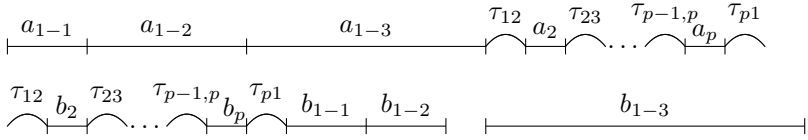



Рис.2: Вид неоптимального расписания $S$.

Это расписание несложно перестроить в расписание длины $\tilde{F}$ следующим образом. Разделим операцию $b_{1-3}$ на два фрагмента, $b'_{1-3}$ и $b''_{1-3}$ так, что $b''_{1-3} = T^* + \sum_{j=2}^{p} a_j$. Фрагмент $b'_{1-3}$ выполним в момент времени 0, сдвинув вправо интервалы выполнения операций машины $M_2$ и, при необходимости, операцию $a_{1-3}$ и последующие операции машины $M_1$. Длина полученного расписания будет равна $\max\{\ell_1 + T^*, \ell_2 + T^*, d_{1-3}\} \le \tilde{F}$. ■

**Задача с заданной базой.** Без ограничения общности считаем, что вершина $v_1$ является базой. Для задачи $ROm|pmtn|F_{\max}$ величина $\bar{F} \doteq \max\{\max \ell_i + T^*, \max\limits_k\{\max\limits_{J_j \in \mathcal{J}_k} d_j + 2\tau_{1k}\}\}$ является нижней оценкой оптимума.

Вершину $v_k$ назовем *перегруженной*, если $\sum\limits_{J_j \in \mathcal{J}_k} d_j \ge \bar{F} - 2\tau_{1k}$.

Справедлива следующая

**Теорема 2.** *Пусть кратчайший обход $R$ графа $G$ может быть найден за время $t_{TSP}$, $v_2$ и $v_p$ — соседние с базой вершины в $R$. Если одна из вершин $v_1$, $v_2$ и $v_p$ является перегруженной, то для задачи $RO2|pmtn, free - depot|F_{\max}$ оптимальное расписание имеет длину $\bar{F}$ и такое расписание с не более чем одним прерыванием может быть найдено за время $O(n + t_{TSP})$.* ■

Доказательство этой теоремы аналогично доказательству теоремы 1.

Работа выполнена при финансовой поддержке РФФИ, грант № 08-01-00370.

**Литература**

1. Пяткин А. В., Черных И. Д. Задача открытого типа с маршрутизацией и разрешением прерываний на двухвершинной сети // Материалы IV Всероссийской конференции “Проблемы оптимизации и экономические приложения”, Омск, 29.06–4.07.09, с. 158.

# МАТЕМАТИКО–КОМПЬЮТЕРНАЯ ОБРАБОТКА КВВ–ЭКСПЕРИМЕНТОВ ПО РАСПОЗНАВАНИЮ ЛЕГОЧНЫХ ЗАБОЛЕВАНИЙ

**Чучалин А. Г.(ФГУ НИИ пульмонологии),**
**Кудрявцев В. Б.(МГУ, мех.–мат. ф–т),**
**Алексеев Д. В. (МГУ, мех.–мат. ф–т),**
**Анаев Э. Х.(ФГУ НИИ пульмонологии),**
**Анохина Т. Н.(МГУ, хим. ф–т),**
**Носов М. В.(МГУ, мех.–мат. ф–т),**
**Ревельский А. И.(МГУ, хим. ф–т),**
**Ревельский И. А. (МГУ, хим. ф–т),**
**Родионов А. А.(МГУ, хим. ф–т)**

Статья посвящена исследованию возможности диагностики хронической обструктивной болезни легких (ХОБЛ) и бронхиальной астмы, основанной на изучении состава среднелетучих органических соединений на ультранизком уровне в конденсате выдыхаемого воздуха (КВВ), с использованием методов выделения примесей и хромато-масс-спектрометрическом анализе всего концентрата аналитов КВВ-экспериментов.

**Введение**

На основании обработки результатов анализа 70 образцов КВВ, собранных у здоровых, больных ХОБЛ и бронхиальной астмой людей, с использованием специально разработанного алгоритма, основанного на линейных методах теории распознавания образов, установлено, что можно различить здоровых и больных бронхиальной астмой с надежностью 75%, здоровых и больных ХОБЛ - с надежностью 85%, больных бронхиальной астмой и ХОБЛ - с надежностью 83%. Установлен ряд веществ, которые могут быть потенциальными маркерами рассматриваемых заболеваний.

После обработки результатов анализа 70 образцов КВВ, собранных у здоровых, больных ХОБЛ и больных бронхиальной астмой, с использованием специально разработанного алгоритма, основанного на линейных методах теории распознавания образов, было установлено, что на основании полученных данных можно различить здоровых и больных бронхиальной астмой с надёжностью 75%, здоровых и больных ХОБЛ — с надёжностью 85%, больных бронхиальной астмой и ХОБЛ — с надёжностью 83%. Из 40 соединений, по которым проводилось сравнение, 9 имеют наибольший информационный вес.

В случае "здоровые - ХОБЛ"это были два неидентифицированных вещества с известными временами удерживания и масс-спектрами (ЭИ), этил цитрат и 2,3-дигидро-1-Н-инден-1-он. В случае "здоровые - бронхиальная астма другие три из неидентифицированных веществ и деканол-1. В случае "бронхиальная астма - ХОБЛ одно неидентифицированное вещество, деканол-1 и 2-феноксиэтанол. Эти вещества могут быть потенциальными маркерами рассмотренных заболеваний. Данные, химического анализа были организованы в таблицы следующего вида:

| $Z$ | $P_1$ | $P_2$ | $\dots$ | $P_S$ |
|---|---|---|---|---|
| $Z_1$ | $z_{1,1}$ | $z_{1,2}$ | $\dots$ | $z_{1,s}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $Z_{n(z)}$ | $z_{n(z),1}$ | $z_{n(z),2}$ | $\dots$ | $z_{n(z),s}$ |

| $B$ | $P_1$ | $P_2$ | $\dots$ | $P_S$ |
|---|---|---|---|---|
| $B_1$ | $b_{1,1}$ | $b_{1,2}$ | $\dots$ | $b_{1,s}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $B_{n(b)}$ | $b_{n(b),1}$ | $b_{n(b),2}$ | $\dots$ | $b_{n(b),s}$ |

| $A$ | $P_1$ | $P_2$ | $\dots$ | $P_S$ |
|---|---|---|---|---|
| $A_1$ | $a_{1,1}$ | $a_{1,2}$ | $\dots$ | $a_{1,s}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $A_{n(a)}$ | $a_{n(a),1}$ | $a_{n(a),2}$ | $\dots$ | $a_{n(a),s}$ |

В данном случае $n(z) = 60; n(a) = n(b) = 40, s = 40$.

**Математический метод распознавания (диагностики).**

Итак, имеется три группы прямоугольных таблиц, которые обозначены $Z, A, B$. Первая группа содержала 2х30=60 строк, вторая — 2х20=40 строк, третья — 2х20=40 строк, число столбцов (признаков) в каждой таблице было равно 40 (по количеству веществ), элементами таблицы являются неотрицательные действительные числа, равные концентрации соответствующего вещества в отдельных образцах. Исследовались пары таблиц $(Z, A), (Z, B), (A, B)$. Для каждой пары таблиц искались наборы признаков и такая линейная комбинация их числовых значений, что на строчках из первой таблицы каждой пары по соответствующим признакам получалось положительное число, а на строчках второй таблицы - отрицательное число; причем суммарное число, в которых не получался требуемый знак, т.е. строк из первой таблицы, на которых линейная комбинация отри-

цательна, и строк из второй таблицы, на которых она положительна, должно быть минимально. Поиск линейной комбинации соответствует прохождению разделяющей плоскости в трехмерном пространстве через начало координат. Такое требование было связано с тем, что концентрации веществ, полученные в результате эксперимента были известны только в относительном ( а не абсолютном выражении) т.е. предоставленные данные представлены были известны с точностью до пропорциональности. Таким образом, точки соответствующие строкам первой таблицы пары, попадают в положительное полупространство (относительно найденной гиперплоскости), а точки второй таблицы - в отрицательное полупространство.

Условия на минимизацию числа признаков, по которым производилась бы диагностика (не более 3–4), ограничение на качество распознавания и возможности по быстродействию вычислительных средств привели к тому, что были использованы выборки по трем признакам. Выборки по двум признакам давали неудовлетворительную точность распознавания ( около 60%), а выборки по 4 и более признакам потребовали бы чрезмерно больших затрат машинного времени и, в случае применении на практике, увеличили бы стоимость проведения соответствующих анализов и сложность процедуры диагностики.

**Формальная математическая постановка задачи**

Представим строки таблиц как вектора $s$–мерного евклидового пространства. Даны множества векторов, обозначим их $Z, A, B \subset \mathbb{R}^s$. Будем обозначать $\Pi_{k,l,m}(x_1, \ldots, x_{40}) = (x_k, x_l, x_m)$ — проекция вектора $(x_1, \ldots, x_s)$ на подпространство, образованное векторами $e_k$, $e_l$ и $e_m$. Аналогично, обозначим проекцию множества $M$ как $\Pi_{k,l,m}(M) = \{\Pi_{k,l,m}(m) \Big| m \in M\}$. Мощность множества $M$ будем обозначать $|M|$.

Было рассмотрено 3 случая:

1. $M_1 = Z, M_2 = B$.
2. $M_1 = Z, M_2 = A$.
3. $M_1 = B, M_2 = A$.

Для каждого случая решалась следующая задача: Пусть заданы множества $M_1$ и $M_2$, мощности которых равны $n_1 = |M_1|$ и $n_2 = |M_2|$. Будем рассматривать всевозможные их проекции $M_1' = M_1(k,l,m) = \Pi_{k,l,m}(M_1)$ и $M_2' = M_2(k,l,m) = \Pi_{k,l,m}(M_2)$ . Для каждой построим "почти"отделяющую гиперплоскость. Дело в том, что

множества могут не быть 0-линейно отделимы (как это было в случае численных данных, полученных из эксперимента), таким образом, рассматривались всевозможные гиперплоскости вида $a_1x_1 + a_2x_2 + a_3x_3 = 0$. Подсчитывались количества неправильно распознанных векторов. Пусть $E_1 = \left|\{(x_1, x_2, x_3) \in M_1' \middle| a_1x_1 + a_2x_2 + a_3x_3 \leq 0\}\right|$ и $E_2 = \left|\{(x_1, x_2, x_3) \in M_2' \middle| a_1x_1 + a_2x_2 + a_3x_3 > 0\}\right|$ — число ошибок распознавания первого и второго множеств. Далее находился минимум $U_{k,l,m} = \min\limits_{a_1,a_2,a_3}(k_1E_1 + k_2E_2)$, где весовые коэффициенты (в данном случае) выбирались обратно пропорционально мощности соответствующего множества, $k_i = 1/n_i$, $i = 1, 2$.

После того находился минимум $U = U(M_1, M_2) = \min\limits_{1\leq k<l<m\leq 40} U_{k,l,m}$. Далее рассматривался вопрос о "устойчивости"полученных решений. Для этого фиксировалась пара чисел $(q_1, q_2)$, случайным образом выбирались подмножества $M_1^{(i)} \subset M_1$ и $M_2^{(i)} \subset M_2$, $i = 1, 2, \ldots, N$, такие, что $|M_1^{(i)}| = |M_1| - q_1$ и $|M_2^{(i)}| = |M_2| - q_2$ и для них решалась та же задача минимизации и находилось $U(M_1^{(i)}, M_2^{(i)})$, после чего находилось среднее значение $U_{cp.}(q_1, q_2)$ по $N = 200$ случайным парам $(M_1^{(i)}, M_2^{(i)})$. Кроме того, подсчитывались частоты, с которыми различные параметры участвуют в оптимальном решении. Более формально, для каждой пары $(M_1^{(i)}, M_2^{(i)})$ строился $w((M_1^{(i)}, M_2^{(i)})) = (w_1, \ldots, w_s)$, следующим образом:

$$w_j = \begin{cases} 1, & \text{при } j = k_{min}, l_{min}, m_{min}; \\ 0, & \text{иначе ,} \end{cases}$$

где $k_{min}, l_{min}, m_{min}$ такие, что $U(M_1^{(i)}, M_2^{(i)}) = U_{k_{min},l_{min},m_{min}}$ Далее, вектора $w((M_1^{(i)}, M_2^{(i)}))$ усреднялись по $(M_1^{(i)}, M_2^{(i)})$ и делились на 3 (для нормировки). Полученные величины - вектор информационные веса признаков - отражают (в некоторой степени) значимость данного параметра для решения задачи разделения множеств. Полученные разделяющие плоскости могут быть использованы в дальнейшем для выработки рекомендаций для диагностирования заболеваний на основе измеренных концентраций.

**Выводы**

1. Задача разделения может быть решена с точностью:
а) на паре "Здоровые - Астма"правильное распознавание - 75%;
б) на паре "Здоровые - ХОБЛ"правильное распознавание - 85%;
в) на паре "ХОБЛ - Астма"правильное распознавание - 84%.
2. Лишь ограниченное количество параметров представляет интерес с точки зрения решения каждой задачи. При решении задачи а) основной информационный вклад дают параметры №№ 2, 6, 1 и 25. При решении задачи б) основной информационный вклад дают параметры №№ 8, 10, 4 и 31. При решении задачи в) основной информационный вклад дают параметры №№ 27, 25 и 1. таким образом значения 9 параметров (из 40) позволяют решить задачу постановки диагноза.
3. Как показывает численный эксперимент, при уменьшении выборки происходит постепенное уменьшение точности распознавания. Поэтому представляется вероятным, что при увеличении объема выборки и точности или достоверности данных возможно увеличение точности распознавания.

**Заключение**

С использованием разработанного способа анализа водных растворов исследован состав смесей органических соединений в 70–ти образцах КВВ, собранных у больных ХОБЛ, бронхиальной астмой и здоровых людей. В результате математической обработки полученных данных показана возможность различать с высокой надёжностью здоровых, больных ХОБЛ и больных бронхиальной астмой между собой. Полученные результаты открывают широкие перспективы для дальнейших исследований.