УДК 519.682.1

С.В. Листровой, д-р техн. наук
Украинский государственный университет
железнодорожного транспорта
(Украина, 61050, Харьков, пл. Фейрбаха, 7,
тел. (050) 9355042, e-mail: om1@ yandex.ru),
А.В. Сидоренко
Научно-производственное предприятие «Стальэнерго»
(Украина, 61105, Харьков, ул. Федоренко 9,
тел. (050) 9800852, e-mail: cdandrey@gmail.com)

Метод решения *k-SAT*-задачи сведением ее к задаче о покрытии

Предложен алгоритм решения *k-SAT*-задачи в среднем за полиномиальное время и 3-*SAT*-задачи за полиномиальное время. Предлагаемый метод позволяет существенно сократить время решения *SAT*-задач.

Запропоновано алгоритм розв'язку *k-SAT*-задачі в середньому за поліноміальний час і 3-*SAT*-задачі за поліноміальний час. Запропонований метод дозволяє істотно скоротити час розв'язку *SAT*-задач .

К л ю ч е в ы е с л о в а : SAT-задача, полиномиальная сводимость.

Решение *SAT*-задач весьма актуально в системах автоматической проверки доказательств, где формулой называют набор клозов, под которыми понимают дизъюнкцию некоторого количества литералов — переменных X и $\overline{X}$. Большое значение эти задачи имеют при выяснении выполнимости схем CIRCUIT-SAT и распознавании образов. Важное место в исследовании *SAT*-задач занимает разработка программ для их решения, называемых *SAT*-солверами. Современные *SAT*-солверы способны быстро решать многие задачи, считавшиеся нерешаемыми несколько лет назад.

Известно много экспоненциальных алгоритмов решения *SAT*-задачи и эвристических подходов полиномиальной сложности к ее решению. Например, в алгоритме Мониена и Шпикермайера для задачи 3-*SAT* использован простой перебор: вместо каждой переменной поочередно выполняется подстановка единицы или нуля и затем рекурсивно решается задача меньшего размера. Этот алгоритм имеет временную сложность $\boldsymbol{O}\,(1{,}84^n)$. Алгоритм решения задачи пропозициональной выполнимости формул в конъюнктивной нормальной форме [1] имеет временную сложность $\boldsymbol{O}\,(1{,}074^n)$.

В общем случае можно выделить два основных типа алгоритмов для решения *SAT*-задач:

1) алгоритмы локального поиска, которые начинаются с какого-либо набора значений (не выполняющие всю формулу); затем их модифицируют, пытаясь последовательно приблизиться к выполняющему набору.

2) так называемые *DPLL*-алгоритмы (соответственно именам создателей: Davis, Putnam, Logemann, Loveland (1968 г.)), обходящие дерево всевозможных наборов и выполняющие поиск в глубину.

Предлагается эффективный алгоритм решения 3-*SAT*-задачи и произвольной *k*-*SAT*-задачи полиномиальной сложности.

Формализация *SAT*-задачи и ее решение. Рассмотрим булеву функцию $F\,(x_1, x_2, ..., x_n)$ в конъюнктивной форме записи:

$$F(x_1, x_2, ..., x_n) = (x_1^{\sigma_{11}} \vee x_2^{\sigma_{12}} \vee ... \vee x_n^{\sigma_{1n}}) \wedge ... \wedge (x_1^{\sigma_{m1}} \vee x_2^{\sigma_{m2}} \vee ... \vee x_n^{\sigma_{mn}}),$$

где

$$x_i^{\sigma} = \begin{cases} x_i, & \sigma = 1, \\ \overline{x_i}, & \sigma = 0; \end{cases}$$

$\vee$ и $\wedge$ — булевы операции, моделирующие простейшие логические высказывания «ИЛИ» и «И». Для любого двоичного набора $x = (x_1, x_2, ..., x_n)$ функция принимает одно из двух возможных значений: 1 или 0. Задача «выполнимость» заключается в ответе на вопрос: существует ли набор значений переменных $x = (x_1, x_2, ..., x_n)$, обращающий функцию F в единицу.

Как показано в [2], *SAT*-задачу можно рассматривать как задачу о покрытии. Для этого по булевой функции построим булеву матрицу $\boldsymbol{B}$, в которой столбцам соответствуют переменные $(X_1, X_2, ..., X_n)$ и $(\overline{X}_1, \overline{X}_2, ...$ $..., \overline{X}_n)$, а строкам — дизъюнкты булевой функции. В общем случае число столбцов в матрице $\boldsymbol{B}$ равно $2n$, а число строк — числу дизъюнктов m в булевой функции. Например,

$$F = (X_1 \vee X_2 \vee X_3)(\overline{X}_1 \vee \overline{X}_2 \vee \overline{X}_3)(X_1 \vee \overline{X}_3)(X_3 \vee \overline{X}_1)(X_1 \vee \overline{X}_2). \quad (1)$$

Перенумеруем дизъюнкты булевой функции:

<table>
<tr><td>1 $(X_1 \vee X_2 \vee X_3)$</td><td colspan="2">2 $(\overline{X}_1 \vee \overline{X}_2 \vee \overline{X}_3)$</td></tr>
<tr><td>3 $(X_1 \vee \overline{X}_3)$</td><td>4 $(X_3 \vee \overline{X}_1)$</td><td>5 $(X_1 \vee \overline{X}_2)$</td></tr>
</table>

.

Тогда матрица **B** примет вид

$$B = \begin{array}{c|cccccc|} & X_1 & X_2 & X_3 & \overline{X}_1 & \overline{X}_2 & \overline{X}_3 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 \\ 3 & 1 & 0 & 0 & 0 & 0 & 1 \\ 4 & 0 & 0 & 1 & 1 & 0 & 0 \\ 5 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}.$$

Столбцы, соответствующие переменным X_j и $\overline{X}_i$ в матрице **B**, будем называть инверсными. Если в матрице **B** существует покрытие строк единицами, принадлежащее неинверсным столбцам, то это значит, что функция F выполнима, если такого покрытия не существует, то она невыполнима. Обозначим переменные $\overline{X}_j$ через Z_j. Тогда матрица **B** для булевой функции (1) примет следующий вид:

$$B = \begin{array}{c|cccccc|} & X_1 & X_2 & X_3 & Z_1 & Z_2 & Z_3 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 \\ 3 & 1 & 0 & 0 & 0 & 0 & 1 \\ 4 & 0 & 0 & 1 & 1 & 0 & 0 \\ 5 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}, \qquad (2)$$

где $Z_j = 0$, если $X_j = 1$, и $Z_j = 1$, если $X_j = 0$.

Как показано в [2], задачу о минимальном покрытии для произвольной матрицы **B**, задаваемой некоторой булевой функцией

$$F = (X_l \vee X_b \vee ... \vee X_k)(X_s \vee X_r \vee ... \vee X_t)(X_q \vee X_d \vee ... \vee X_h),$$

можно рассматривать как задачу нахождения минимального набора переменных $\{X_i = 1\}$, при которых булева функция (1) выполнима. Эту задачу можно записать в виде $\min_j \{X_j = 1\}$ при выполнении ограничений

$$(X_l \vee X_b \vee ... \vee X_k)(X_s \vee X_r \vee ... \vee X_t)(X_q \vee X_d \vee ... \vee X_h) = 1.$$

Переходя к двойственной булевой функции (в дизъюнктивной нормальной форме (ДНФ)), получаем $\min_j \{X_j = 0\}$ при

$$X_l X_b ... X_k \vee X_s X_r ... X_t \vee ... \vee X_q X_d ... X_h = 0, \qquad (3)$$

откуда следует, что задачу о наименьшем покрытии можно рассматривать как задачу нелинейного булевого программирования, которая заключается в нахождении наименьшего числа переменных $\{X_j = 0\}$, обращающих в нуль левую часть ограничения (3). Если существует хотя бы одно покрытие, то необходимо выяснить, существует ли хотя бы один набор переменных $\{X_j = 0\}$, обращающих в нуль левую часть ограничения (3). Для матрицы (2) условие существования хотя бы одного покрытия имеет вид

$$X_l Z_b \dots X_k \vee X_s Z_r \dots X_t \vee \dots \vee X_q Z_d \dots Z_h = 0 \qquad (4)$$

при выполнении следующих условий:

$$\begin{aligned} &\text{если } X_j = 1, \text{ то } Z_j = 0, \\ &\text{если } X_j = 0, \text{ то } Z_j = 1. \end{aligned} \qquad (5)$$

Таким образом, для установления выполнимости булевой функции требуется установить наличие хотя бы одного покрытия строк единицами в матрице ***B*** (2), удовлетворяющее условию (5). Для этого необходимо решить нелинейное булево уравнение (4), и решение этого уравнения должно удовлетворять условию (5). Обозначив произвольный дизъюнкт в (4) через S_i, а число дизъюнктов — через m, задачу (4), (5) можно записать в виде

$$\sum_{i=1}^{m} S_i = 0. \qquad (6)$$

Для решения задачи (5), (6) переменные булевой функции

$$x_j^{\sigma} = \begin{cases} x_j, & \sigma = 1, \\ z_j, & \sigma = 0, \end{cases}$$

принимающие значения X_j, Z_j, будем характеризовать весовыми характеристиками h_j, определяющими частоту появления переменных X_j и Z_j в слагаемых уравнения (6). При этом каждый дизъюнкт S_i также будем характеризовать весовой характеристикой p_i, равной сумме частот переменных h_j, образующих данное слагаемое. Под преобразованием некоторой булевой функции F_r в F_{r+1} будем подразумевать изменение функции F_r в результате подстановки в нее пар $X_j = 1$, $Z_j = 0$ или $X_j = 0$, $Z_j = 1$. Поскольку в функции F_r в результате подстановки $X_j = 1$ или $Z_j = 1$ появляются дизъюнкты с меньшим числом переменных, будем выполнять операцию поглощения вида $XZ \vee X = X$. Если в результате подстановки в нее пар $X_j = 1$, $Z_j = 0$ или $X_j = 0$, $Z_j = 1$ вытекает необходимость выполнения равенства $X_j + Z_j = 0$ (т.е. (6) принимает вид 1 = 0), то будем считать, что возникло противоречие и функция F_r невыполнима. Если в процессе

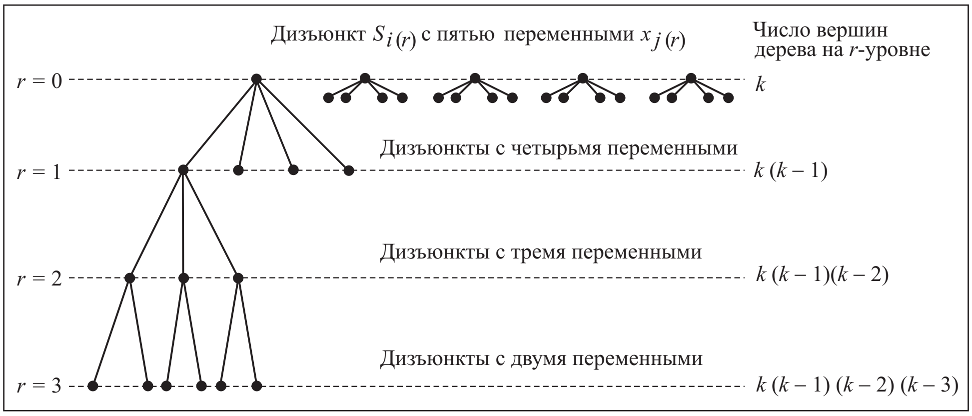

Рис. 1. Дерево формирования булевых функций F_r

преобразования появляются дизъюнкты, состоящие из одной переменной, то эти переменные полагаем равными нулю и заносим их в решение.

Основная идея решения уравнения (6) — выбор некоторого дизъюнкта в исходной булевой функции F и проверка возможности обнуления всех остальных слагаемых при обнулении поочередно переменных, принадлежащих выбранному дизъюнкту. При этом образуются некоторые промежуточные функции F_r, в которых также выделяем дизъюнкты и пытаемся обнулить все слагаемые на основе выбора одного дизъюнкта в этих функциях, полагая переменные в выбранных дизъюнктах поочередно равными нулю. Эта процедура продолжается до тех пор, пока не получим тождество $0 = 0$ или прейдем к противоречию $1 = 0$. В первом случае исследуемая функция выполнима, а во втором — невыполнима. В общем случае множество таких функций F_r, сформированных на основе одного дизъюнкта, выбранного в исходной булевой функции F, можно представить в виде дерева.

На рис. 1 представлено дерево всех F_r, которые придется построить для решения задачи k-SAT при $k = 5$. Для построения процедуры формирования F_r по дереву введем следующие обозначения: $S_{i(r)}$ — i-й дизъюнкт на r-м уровне в дереве; $j(r)$ — номер переменной в дизъюнкте $S_{i(r)}$ на r-м уровне в дереве; $x_{j(r)}$ — переменная в дизъюнкте $S_{i(r)}$ под номером $j(r)$. В соответствии с правилами преобразования булевых функций F_r с учетом введенных обозначений рассмотрим следующую процедуру решения уравнения (6) при выполнении условий (5) для решения задачи k-SAT, т.е. задачи выполнимости, в каждом дизъюнкте которой содержится по k-переменных.

Процедура А.

Ш а г 1. Выбираем в (6) дизъюнкт $S_{i(r)}$ с минимальным числом переменных и наибольшей суммарной весовой характеристикой p_r; выбираем в нем переменную $x_{j(r)}$ с наибольшим значением частоты h_j (что соответствует нулевому уровню $r = 0$) и переходим к следующему шагу.

Ш а г 2. Присваиваем переменной $x_{j(r)}$ дизъюнкта $S_{i(r)}$ нулевое значение и выполняем преобразование булевой функции F_r в F_{r+1}, переходим к следующему шагу.

Ш а г 3. Проверяем функцию F_{r+1} на невыполнимость и проверяем, есть ли в дизъюнкте $S_{i(r)}$ переменные, которые не были приравнены нулю, т.е. проверяем, не вытекает ли из процесса преобразования F_r в F_{r+1} противоречие вида $X_i + Z_i = 0$. Если да, то переходим к шагу 2, иначе — к следующему шагу.

Ш а г 4. Проверяем, выполнима или нет функция F_{r+1} (т.е. выполнение тождества $0 = 0$ в уравнении (6)). Если да, то алгоритм заканчивает работу, иначе — переходим к следующему шагу.

Ш а г 5. Проверяем функцию F_{r+1} на невыполнимость: возникает противоречие вида $X_i + Z_i = 0$ при попытке обращения F_{r+1} в нуль. Если функция не обращена в нуль и при этом противоречия не возникали, то переходим к следующему шагу, если противоречие возникло (т.е. она невыполнима), то переходим к шагу 7.

Ш а г 6. Переходим на следующий уровень. Для этого находим дизъюнкт $S_{i(r)}$ с минимальным числом переменных и наибольшей суммарной весовой характеристикой p_r, выбираем в нем переменную $x_{j(r)}$ с наибольшим значением частоты h_j в булевой функции F_r и переходим к шагу 2.

Ш а г 7. Проверяем, проверены ли на нулевом уровне ($r = 0$) все переменные на возможность обнуления дизъюнктов в уравнении (6). Если да, то алгоритм заканчивает работу, так как функция невыполнима, иначе — переходим к следующему шагу.

Ш а г 8. Возвращаемся на предыдущий ($r - 1$)-й уровень ветвления и переходим к шагу 2.

Блок-схема данного алгоритма представлена на рис. 2. После выбора дизъюнкта $S_{i(r)}$ в исходной булевой функции F процедура ***А*** анализирует не всю исходную булеву функцию F, а лишь ту часть дизъюнктов, с которыми по переменным пересекается выбранный дизъюнкт. Последовательности дизъюнктов булевой функции, которые не пересекаются по переменным, входящим в дизъюнкты, образующие эти последовательности, будем называть, независимыми последовательностями дизъюнктов. Ясно, что если в булевой функции число переменных равно $2n$, а число переменных в каждом дизъюнкте равно k, то максимальное число

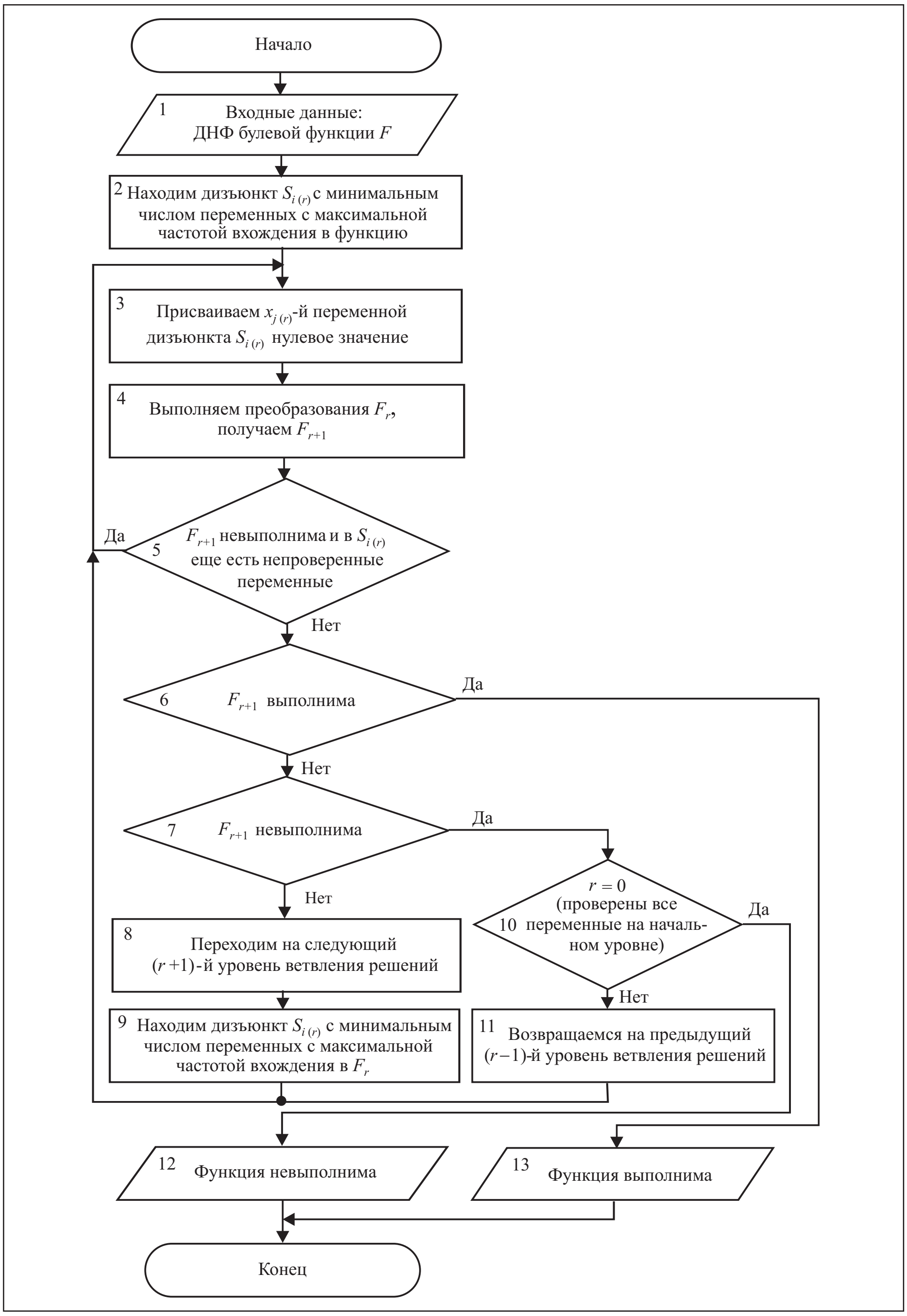

Рис. 2. Блок-схема реализации процедуры ***A***

таких независимых последовательностей не может превысить величину $2n/k$, т.е. в худшем случае на основе предложенной процедуры в исходной булевой функции придется обнулять не более $2n/k$ таких последовательностей и, следовательно, анализировать не более $2n/k$ деревьев.

Оценим сложность анализа одного такого дерева. В соответствии с построенным деревом число функций F_r, которые придется преобразовывать, равно суммарному числу вершин на всех уровнях анализируемого дерева. Как следует из рис. 1, число вершин на нулевом уровне будет равно k, на первом уровне — $k(k-1)$, на третьем — $k(k-1)(k-2)$ и так далее. Следовательно, число вершин на нулевом уровне равно k, на втором — число вершин не превысит k^2, на третьем — оно не превысит k^3 и на последнем — не превысит k^{k-1}.

Рассмотрим сумму $k+k^2+k^3+...+k^{k-1}$. Ясно, что она не может превысить величину k^k. Например, для задачи «3-выполнимость» в дереве придется преобразовать в худшем случае $k+k(k-1)+k(k-1)(k-2)=15=$ $=3+3(3-1)+3(3-1)(3-2)=15$ функций. Учитывая, что при решении *k-SAT*-задачи придется анализировать $2n/k$ деревьев, в худшем случае придется преобразовать $2k^{k-1}n$ функций F_r. В случае решения 3-*SAT*-задачи это число не превысит величину $\frac{15\cdot 2n}{3}=10n$.

Для анализа исходного уравнения и произвольной булевой функции F_r в соответствии с предложенной процедурой $\boldsymbol{A}$ потребуется u различных операций. Число u определяется $2mn$ операциями сравнения для определения частот h_j появления каждой переменной в выбранном слагаемом и остальных слагаемых плюс mk операциями сложения для определения суммарного значения весовой характеристики p_i каждого слагаемого в уравнении (6) и плюс $m\log_2 m$ операциями сравнения выбора максимального элемента в массиве из m элементов, т.е. $u=2mn+mk+m\log_2 m$. В худшем случае на каждом шаге процедуры $\boldsymbol{A}$ будет обнулятся только одно слагаемое, т.е. в уравнении (6) после первого шага останется $m-1$ слагаемое, далее $m-2$, $m-3$, ..., 1 слагаемых, т.е. обнуление всех слагаемых процедуры $\boldsymbol{A}$ в худшем случае потребует выполнения $\frac{m(m+1)}{2}u$ операций.

Таким образом, общее число элементарных операций при решении *k-SAT*- задачи в худшем случае не может превысить величину

$$O\left(\frac{2k^{k-1}nm(m+1)}{2}u\right)=O\,(k^{k-1}mn\,(m+1)(2mn+mk+m\log_2 m))\approx$$

$$\approx O\left(2m^3n^2k^{k-1}\left(1+\frac{k+\log_2 m}{2n}\right)\right). \qquad (7)$$

При решении 3-*SAT*-задачи число элементарных операций в худшем случае не может превысить величину

$$O\left(10m^2(m+1)\,n^2\left(1+\frac{3+\log_2 m}{2n}\right)\right)\approx O\left(10m^3n^2\left(1+\frac{3+\log_2 m}{2n}\right)\right). \qquad (8)$$

Известно, что *k*-*SAT*-задача может быть сведена к 3-*SAT*-задаче за полиномиальное время. В частности, если дизъюнкт S_i содержит более трех литералов, например $S_i=(\lambda_1+\lambda_2+\ldots+\lambda_k)$, $k>3$, то можно заменить S_i на $k-2$ дизъюнктов [3] :

$$S_i=(\lambda_1+\lambda_2+x_1)(\overline{x_1}+\lambda_3+x_2)(\overline{x_2}+\lambda_4+x_3)\cdot\ldots\cdot(\overline{x_{k-3}}+\lambda_{k-1}+\lambda_k), \qquad (9)$$

где $x_1,\ldots,x_{k-3}$ — новые переменные. Набор этих новых дизъюнктов выполним тогда и только тогда, когда выполним дизъюнкт S_i. Таким образом, при переходе от *k*-*SAT*-задачи к задаче 3-*SAT* каждый дизъюнкт *k*-*SAT*-задачи увеличивает число переменных в булевой функции на $(k-3)$, а число дизъюнктов — на $k-2$ дизъюнкта. Следовательно, после сведения число переменных и число дизъюнктов в новой задаче будут соответственно $n'=(n+m(k-3))$ и $m'=(k-2)\,m$. Подставляя в (8) n' и m' в качестве новых значений n и m, получаем оценку сложности решения *k*-*SAT*-задачи с помощью процедуры ***A***:

$$O\left(10(k-2)^3m^3(n+m(k-3))^2\left(1+\frac{3+\log_2(k-2)+\log_2 m}{2(n+m(k-3))}\right)\right). \qquad (10)$$

Предложенная процедура ***A*** решения *k*-*SAT*-задачи представляет собой формально полиномиальный алгоритм, но с высокой степенью полинома (7), и, следовательно, реализуется за экспоненциальное время, при этом решение 3-*SAT*-задачи процедура ***A*** осуществляет за полиномиальное время. Следовательно, преобразование *k*-*SAT*-задачи за полиномиальное время в 3-*SAT*-задачу приводит к алгоритму полиномиальной сложности. При решении *k*-*SAT*-задачи с помощью процедуры ***A*** без сведения ее к 3-*SAT*-задаче алгоритм преобразования имеет экспоненциальную сложность. При этом выбор на каждом шаге процедуры ***A*** дизъюнкта $S_{i(r)}$ с минимальным числом переменных и наибольшей суммарной весовой характеристикой p_r позволяют существенно сократить среднее время реализации алгоритма посредством обнуления максимально возможного числа слагаемых на каждом шаге процедуры.

Таким образом, процедура ***A*** представляет собой полный перебор вариантов возможного обнуления дизъюнктов. В случае выполнимости функции обнуление происходит за полиномиальное время, а в случае ее невыполнимости после первой попытки обнуления либо первого либо m-го дизъюнкта выясняется невозможность его обнуления и в дереве происходит быстрый отбор функций, которые необходимо анализировать. В связи с этим процедура $\boldsymbol{A}$ позволяет решить k-SAT-задачу в среднем за полиномиальное время. В случае преобразования k-SAT-задачи в 3-SAT-задачу она решается с помощью процедуры $\boldsymbol{A}$ за полиномиальное время, определяемое соотношением (10).

Рассмотрим примеры работы процедуры $\boldsymbol{A}$ при определении выполнимости булевых функций. Пусть требуется определить выполнимость следующей булевой функции с четырьмя переменными и тринадцатью дизъюнктами:

$$F(x)=(x_1^- + x_0^- + x_3^-)(x_1 + x_0 + x_2)(x_0^- + x_3^- + x_2^-)(x_3^- + x_2^- + x_1)\times$$
$$\times(x_1 + x_0 + x_3)(x_1^- + x_0^- + x_2)(x_3^- + x_0 + x_2)(x_1^- + x_2^- + x_0)(x_0^- + x_3 + x_2)\times$$
$$\times(x_2^- + x_1 + x_0)(x_0^- + x_3^- + x_1)(x_3^- + x_1 + x_2)(x_0^- + x_2^- + x_3), \qquad (11)$$

где полагаем, что знак плюс — это знак логического сложения. Запишем исходное уравнение :

$$F_{r=0} = Z_1Z_0Z_3 + x_1x_0x_2 + Z_0Z_3Z_2 + Z_3Z_2x_1 + x_1x_0x_3 + Z_1Z_0x_2 + Z_3x_0x_2 +$$
$$+ Z_1Z_2x_0 + Z_0x_3x_2 + Z_2x_1x_0 + Z_0Z_3x_1 + Z_3x_1x_2 + Z_0Z_2x_3 = 0. \qquad (12)$$

Определяем частоты h_j появления переменных в каждом слагаемом:

$x_{j(r)}$	x_0	x_1	x_2	x_3	Z_0	Z_1	Z_2	Z_3
h_j	5	5	5	3	6	3	5	6

.

Выбираем дизъюнкт $Z_0Z_3x_1$ с максимальным значением суммы частот h_j $6 + 6 + 5 = 17$, полагаем $Z_0 = 0, x_0 = 1$ и осуществляем преобразование $F_{r=0}$ в $F_{r=1}$. В результате исходное уравнение (12) принимает вид

$$x_1x_2 + Z_3Z_2x_1 + x_1x_3 + Z_3x_2 + Z_1Z_2 + Z_2x_1 + Z_3x_1x_2 = 0. \qquad (13)$$

Проводим операцию поглощения: здесь x_1x_2 поглощает слагаемое $Z_3x_1x_2$, а слагаемые Z_2x_1 и Z_3x_2 поглощают соответственно слагаемые $Z_3Z_2x_1$ и $Z_3x_1x_2$. После этого уравнение (13) будет иметь вид

$$F_{r=1} = x_1x_2 + x_1x_3 + Z_3x_2 + Z_1Z_2 + Z_2x_1 = 0 \,. \qquad (14)$$

Проверяем, обращена ли в нуль функция $F_{r=1}$, т.е. выполнима или нет. В данном случае функция $F_{r=1}$ не обращена в нуль и в процессе преобразования функции $F_{r=0}$ в $F_{r=1}$ противоречия не возникло. Следовательно, переходим на следующий $r+1$ уровень. Снова определяем частоты h_j появления переменных в каждом слагаемом:

$x_{j(r)}$	x_1	x_2	x_3	Z_1	Z_2	Z_3
h_j	3	1	1	1	2	1

. (15)

Далее, среди дизъюнктов, содержащих минимальное число переменных, а такими являются дизъюнкты x_1x_2, x_1x_3, Z_3x_2, Z_1Z_2, Z_2x_1, выбираем дизъюнкт с наибольшим значением весовой характеристики p_j. Согласно (15) весовые характеристики этих слагаемых равны соответственно 4, 4, 2, 3, 5, поэтому выбираем дизъюнкт Z_2x_1. В нем переменной с наибольшей частотой, равной трем, является переменная x_1, поэтому полагаем $x_1=0$, $Z_1=1$. Тогда уравнение (14) принимает вид

$$F_{r=2} = Z_3x_2 + Z_2 = 0. \tag{16}$$

В (16) нельзя сделать поглощений, но в нем появилось слагаемое с одной переменной, поэтому полагаем $Z_2=0$, $x_2=1$ и, следовательно, записываем (16) в виде $Z_3=0$. Таким образом, набор из переменных $Z_0=0$, $x_1=0$, $Z_2=0$, $Z_3=0$ или $\bar{x}_0x_1\bar{x}_2\bar{x}_3$ обращает уравнение (12) в тождество и, следовательно, является выполняющим набором переменных для булевой функции (11).

Рассмотрим процесс работы процедуры ***A*** в случае, когда булева функция невыполнима. Пусть задана функция

$$\begin{gathered}F(x)=(x_0+x_1+\bar{x}_2)(\bar{x}_3+\bar{x}_0+x_2)(x_0+\bar{x}_3+x_2)(x_1+\bar{x}_2+\bar{x}_0)\times\\ \times(\bar{x}_0+\bar{x}_1+x_2)(x_3+x_1+\bar{x}_0)(x_3+x_0+\bar{x}_2)(x_0+\bar{x}_2+\bar{x}_1)(x_3+x_0+x_2)\times\\ \times(x_3+x_0+x_1)(x_3+\bar{x}_2+\bar{x}_0)(\bar{x}_2+\bar{x}_3+\bar{x}_1)(x_1+\bar{x}_2+\bar{x}_3).\end{gathered}\tag{17}$$

Записываем исходное уравнение:

$$\begin{gathered}F_{r=1}=x_0x_1Z_2+Z_3Z_0x_2+x_0Z_3x_2+x_1Z_2Z_0+Z_0Z_1x_2+x_3x_1Z_0+x_3x_0Z_2+\\ +x_0Z_2Z_1+x_3x_0x_2+x_3x_0x_1+x_3Z_2Z_0+Z_2Z_3Z_1+x_1Z_2Z_3=0.\end{gathered}\tag{18}$$

Определяем частоту h_j появления переменных в каждом слагаемом (18):

$x_{j(r)}$	x_0	x_1	x_2	x_3	Z_0	Z_1	Z_2	Z_3
h_j	6	5	4	4	5	3	6	4

.

Выбираем дизъюнкт $x_0x_1Z_2$ с максимальным значением суммы частот h_j 6 + 5 + 6 = 17. Выбираем в нем переменную x_0 с максимальным значением h_j и полагаем $x_0 = 0$, $Z_0 = 1$. Тогда исходное уравнение (18) записываем в виде

$$Z_3x_2 + x_1Z_2 + Z_1x_2 + x_3x_1 + x_3Z_2 + Z_2Z_3Z_1 + x_1Z_2Z_3 = 0. \tag{19}$$

В (19) слагаемое x_1Z_2 поглощает слагаемое $x_1Z_2Z_3$ и уравнение (19) принимает вид

$$F_{r=1} = Z_3x_2 + x_1Z_2 + Z_1x_2 + x_3x_1 + x_3Z_2 + Z_2Z_3Z_1 = 0. \tag{20}$$

Поскольку функция $F_{r=1}$ не обращается в нуль и при этом противоречия не возникает, определяем частоту h_j появления переменных в каждом дизъюнкте (20):

$x_{j(r)}$	x_1	x_2	x_3	Z_1	Z_2	Z_3
h_j	2	2	2	2	3	2

.

В (20) дизъюнктами с максимальным суммарным весом, равным пяти, являются дизъюнкты x_1Z_2 и x_3Z_2. Поэтому для дальнейшего анализа выбираем, например, дизъюнкт $S_1 = x_1Z_2$ и в нем выбираем переменную Z_2 с максимальной частотой появления в дизъюнктах функции (20), равной трем. Далее, полагая $Z_2 = 0$, $x_2 = 1$, преобразуем (20) к виду

$$F_{r=2} = Z_3 + Z_1 + x_3x_1 = 0\ . \tag{21}$$

Поскольку в (21) поглощений сделать нельзя и появились слагаемые, содержащие по одной переменной, полагаем $Z_3 = 0$, $x_3 = 1$ и $Z_1 = 0$, $x_1 = 1$, в результате чего получаем 1 = 0, т.е. возникает противоречие. Поэтому возвращаемся на предыдущий уровень $F_{r=1}$ и переходим к попытке обнуления дизъюнкта x_1Z_2 на основе переменной x_1. Для этого в (21) полагаем $x_1 = 0$, $Z_1 = 1$ и получаем

$$F_{r=2} = Z_3x_2 + x_2 + x_3Z_2 + Z_2Z_3 = 0. \tag{22}$$

В (22) x_2 поглощает слагаемое Z_3x_2. Тогда получаем следующее уравнение:

$$F_{r=3} = x_2 + x_1Z_2 + Z_2Z_1 + x_1Z_2 = 0. \tag{23}$$

Поскольку в (23) появились слагаемые, содержащие по одной переменной, полагаем $x_2 = 0$, $Z_2 = 1$ и получаем $x_1 + Z_1 = 0$, т.е. возникло противоречие. Следовательно, обнулить дизъюнкт $S_1 = x_1Z_2$, используя переменные x_1 и Z_2, без возникновения противоречия невозможно. Поэтому возвращаемся на нулевой уровень и проверяем, все ли переменные проверены на возможность обнуления дизъюнкта $x_0x_1Z_2$. В данном случае непроверенными остались переменные x_1 и Z_2. При этом переменная Z_2 имеет большую частоту появления в дизъюнктах, чем x_1. Поэтому, полагая $Z_2 = 0$, $x_2 = 1$, приводим уравнение (18) к виду

$$F_{r=1} = Z_3Z_0 + x_0Z_3 + Z_0Z_1 + x_3x_1Z_0 + x_3x_0 + x_3x_0x_1 = 0\,. \tag{24}$$

В (24) слагаемое x_3x_0 поглощает слагаемое $x_3x_0x_1$. Тогда получаем

$$F_{r=2} = Z_3Z_0 + x_0Z_3 + Z_0Z_1 + x_3x_1Z_0 + x_3x_0 = 0. \tag{25}$$

Функция $F_{r=2}$ не обращается в нуль, и при этом противоречия не возникает. Поэтому определяем частоту h_j появления переменных в каждом дизъюнкте (25):

$x_{j(r)}$	x_0	x_1	x_3	Z_0	Z_1	Z_3
h_j	2	1	2	3	1	2

.

Среди дизъюнктов с двумя переменными выбираем дизъюнкт с максимальной суммарной частотой. Таким является слагаемое Z_3Z_0 с весовой характеристикой, равной пяти. Выбираем в нем переменную Z_0 с наибольшим весом, равным четырем, и полагаем $x_0 = 1$ и $Z_0 = 0$. Тогда (25) принимает вид $Z_3 + x_3 = 0$, т.е. возникло противоречие. Поэтому возвращаемся на предыдущий уровень для обнуления дизъюнкта Z_3Z_0 на основе переменной Z_3. Для этого в (25) полагаем $x_3 = 1$, $Z_3 = 0$ и получаем

$$F_{r=1} = Z_0Z_1 + x_1Z_0 + x_0 = 0. \tag{26}$$

В (26) возникло слагаемое, состоящее из одной переменной. Поэтому полагаем $x_0 = 0$, $Z_0 = 1$ и получаем $Z_1 + x_1 = 0$, т.е. возникло противоречие. Далее, проверяем, все ли переменные $x_{j(r)}$ на нулевом уровне были использованы для обнуления дизъюнкта $x_0x_1Z_2$. В данном случае осталась непроверенной только одна переменная x_1, поэтому полагаем $x_1 = 0$, $Z_1 = 1$. Тогда уравнение (18) принимает вид

$$F_{r=1} = Z_3Z_0x_2 + x_0Z_3x_2 + Z_0x_2 + x_3x_0Z_2 + x_0Z_2 + x_3x_0x_2 + x_3Z_2Z_0 + Z_2Z_3 = 0.$$

После поглощения слагаемым x_0Z_2 слагаемых $Z_3Z_0x_2$ и $x_3x_0Z_2$ получаем

$$F_{r=1} = x_0Z_3x_2 + Z_0x_2 + x_0Z_2 + x_3x_0x_2 + x_3Z_2Z_0 + Z_2Z_3 = 0. \tag{27}$$

Функция $F_{r=1}$ не обращается в нуль, и при этом противоречия не возникает. Поэтому определяем частоту h_j появления переменных в каждом дизъюнкте (27):

$x_{j(r)}$	x_0	x_2	x_3	Z_0	Z_2	Z_3
h_j	3	2	2	2	3	2

.

Среди дизъюнктов с двумя переменными в (27) выбираем дизъюнкт с максимальной суммарной частотой. Таким является слагаемое x_0Z_2 с весовой характеристикой, равной шести. Выбираем в нем переменную x_0 с частотой появления в других дизъюнктах (27), равной трем, и полагаем $x_0 = 0$ и $Z_0 = 1$. Тогда (27) принимает вид

$$Z_3x_2 + x_2 + x_3Z_2 + Z_2Z_3 = 0. \qquad (28)$$

В (28) x_2 поглощает Z_3x_2 , поэтому получаем

$$F_{r=2} = x_2 + x_3Z_2 + Z_2Z_3 = 0. \qquad (29)$$

В (29) появилось слагаемое с одной переменной x_2. Поэтому полагаем $x_2 = 0$, $Z_2 = 1$ и получаем $x_3 + Z_3 = 0$, т.е. возникло противоречие. Поэтому возвращаемся на уровень $r - 1$, т.е. переходим к функции (27) и предпринимаем попытку обнуления дизъюнкта x_0Z_2 на основе переменной Z_2. Для этого, полагая в (27) $Z_2 = 0$, $x_2 = 1$, получаем $Z_3Z_0 + x_0Z_3 + Z_0 + x_3x_0 = 0$. Проведя поглощение, получаем $x_0Z_3 + Z_0 + x_3x_0 = 0$, где появилось слагаемое с одной переменной Z_0. Поэтому полагаем $Z_0 = 0$, $x_0 = 1$ и получаем $Z_3 + x_3 = 0$, т.е. образовалось противоречие. Поэтому возвращаемся на $r - 1$ уровень (в данном случае нулевой) и проверяем, все ли переменные использованы для обнуления дизъюнкта $x_0x_1Z_2$. В данном случае это — последняя переменная. Следовательно, процедура заканчивает работу, так как при равенстве нулю переменных в дизъюнкте $x_0x_1Z_2$ обратить уравнение (18) в тождество невозможно, значит функция (17) невыполнима.

Следует заметить, что в наихудшем случае оценка сверху временной сложности процедуры $\boldsymbol{A}$ является достаточно грубой, так как предполагалось, что при присвоении $X_j = 0$ и $Z_j = 1$ на каждом шаге обнуляется только одно слагаемое, хотя на самом деле обнуляется h_j слагаемых. Поэтому представляет интерес получить среднюю величину оценки временной сложности процедуры $\boldsymbol{A}$.

При проведении эксперимента $2n$ переменных $X_j = 1$ и Z_j размещались по дизъюнктам по равномерному закону распределения. Результаты экспериментальных исследований получены с доверительной вероятностью 0,95. Для получения среднего значения числа элементарных операций (ЭО), выполняемых процедурой $\boldsymbol{A}$, на каждую точку генерировалось от 50 до 70 булевых функций исследуемой размерности. Число дизъюнктов m менялось от 100 до 10000, а число переменных n — от 90 до 2400. Результаты экспериментальных исследований приведены в табл. 1—3.

Из табл. 1—3 видно, что наибольшее число операций процедура выполняет при значениях $\alpha = n / m$, близких к 0,24. При равномерном законе распределения переменных X_i и $\overline{X}_i$ по дизъюнктам при $\alpha < 0{,}24$ булевы функции, как правило, являются невыполнимыми, а при $\alpha > 0{,}24$ —

Таблица 1

n	Число ЭО	$\alpha = n/m$	Выполнимость булевой функции	Время решения, мс	
				минимальное	максимальное
m = 500; k = 3					
100	$1{,}06 \cdot 10^8$	0,2	–	5,06	793 934
110	$7{,}2 \cdot 10^8$	0,22	–	5,06	793 934
120	$8{,}6 \cdot 10^9$	0,24	–	5,06	793 934
130	$4{,}2 \cdot 10^8$	0,26	+	5,06	793 934
140	63 209	0,28	+	5,06	793 934
150	52 096	0,3	+	5,06	793 934
400	286 987	1,25	–	36,5	849
800	10^6	0,625	–	36,5	849
1200	$2{,}24 \cdot 10^6$	0,416	–	36,5	849
1600	$4 \cdot 10^6$	0,313	+	36,5	849
2000	$6{,}2 \cdot 10^6$	0,25	+	36,5	849
2400	$8{,}9 \cdot 10^6$	0,208	+	36,5	849
m =10000; k = 3					
100	324 459	0,2	–	44	819 261
200	$1{,}25 \cdot 10^6$	0,22	–	44	819 261
300	$4{,}89 \cdot 10^6$	0,24	–	44	819 261
400	$9{,}5 \cdot 10^7$	0,26	+	44	819 261
500	$2{,}36 \cdot 10^9$	0,28	+	44	819 261
5000	$6{,}1 \cdot 10^7$	0,3	+	44	819 261

Примечание: знак минус означает невыполнимость булевой функции, а знак плюс — ее выполнимость.

выполнимыми. При значении α, близком к 0,24, вероятность появления выполнимых и невыполнимых булевых функций становится одинаковой и возрастает число операций, выполняемых процедурой $\boldsymbol{A}$. С увеличением значения k (см. табл. 2), когда слагаемое $\frac{k+\log_2 m}{2n}$ в (10) значительно меньше единицы, наблюдается уменьшение временной сложности работы процедуры, что обусловлено возрастанием числа обнуляемых слагаемых на каждом шаге ее работы. Однако, когда указанное в (10) слагаемое становится существенно больше единицы, наблюдается дальнейшее возрастание числа элементарных операций, выполняемых процедурой $\boldsymbol{A}$.

Для тестирования разработанного алгоритма использовано 900 тестов специализированной библиотеки SAT Live [4], каждый из которых содер-

Таблица 2

m	Число ЭО	$\alpha = n/m$	Выполнимость булевой функции	Время решения, мс	
				минимальное	максимальное
n =90; k = 3					
100	16 029	0,9	+	2,15	38 910
200	26 994	0,4	+	2,15	38 910
300	37 413	0,3	+	2,15	38 910
400	$4{,}39 \cdot 10^7$	0,23	+	2,15	38 910
500	$2{,}42 \cdot 10^7$	0,18	–	2,15	38 910
600	$1{,}86 \cdot 10^7$	0,15	–	2,15	38 910

Таблица 3

k	Число ЭО	Выполнимость булевой функции	Время решения, мс	
			минимальное	максимальное
m =500; n =90; α = 18				
3	$2{,}42 \cdot 10^7$	–	7,57	160
10	60 807	+	7,57	160
20	48 555	+	7,57	160
30	55 488	+	7,57	160
40	64 268	+	7,57	160
50	74 508	+	7,57	160

Таблица 4

Тест	Число пере-менных	Число дизъюнк-тов	Среднее число выполненных операций при решении	Время решения, мс		
				среднее	мини-мальное	макси-мальное
CBS_k3_n100_m403_b0	100	400				
. . . .			$51\cdot10^6$	4578	2,28	77843
CBS_k3_n100_m403_b900	100	400				

жит 403 дизъюнкта и 100 переменных с параметром $\alpha = 0{,}248$. Тесты сформированы посредством псевдо-случайного заполнения дизъюнктов. Тестирование осуществлялось на компьютере ASER с Intel Pentium processor T4400, 2,2 Ghz, 3 GB Memory. Результаты тестирования приведены в табл. 4.

Если в процессе работы процедуры $\boldsymbol{A}$ выбранный дизъюнкт невозможно обнулить без возникновения противоречия, то, значит, функция невыполнима, и проводить проверку возможности обнуления остальных слагаемых в (10) не имеет смысла. Поэтому, как свидетельствуют результаты экспериментального исследования, для выяснения факта невыполнимости булевых функций процедурой $\boldsymbol{A}$ требуется существенно меньше времени, чем затрачивается на решение задачи, когда функция выполнима. Это хорошо видно при равномерном распределении переменных X_i и $\overline{X}_i$ по дизъюнктам булевой функции для значений параметра α, значительно меньших 0,24. Также быстро осуществляется решение и для выполнимых функций при значениях параметра α, существенно превышающих 0,24. Наихудший случай для работы процедуры, — когда параметр α принимает значение, близкое к 0,24. В этом случае число элементарных операций, выполняемых процедурой $\boldsymbol{A}$, наиболее близко к верхней оценке, определяемой соотношениями (7), (8).

Выводы

Предложенная процедура $\boldsymbol{A}$ позволяет решать 3-*SAT*-задачу и *k-SAT*-задачу за полиномиальное время. Разработанный метод решения *SAT*-задач позволит создать быстродействующие программы — *SAT*-солверы, способные решать многие задачи дискретной оптимизации большой размерности в масштабе реального времени, ранее считавшиеся нерешаемыми.

СПИСОК ЛИТЕРАТУРЫ

1. *Hirsch E.A.* New Worst-Case Upper Bounds for SAT // Journal of Automated Reasoning. — 2000. — Vol. 24, No 4. — P. 397—420.
2. *Листровой С.В., Минухин С.В.* Метод решения задач о минимальном вершинном покрытии в произвольном графе и задачи о наименьшем покрытии // Электрон. моделирование. — 2012. — **34**, №1.— С. 29—43.
3. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация. Алгоритмы и сложность. — М.: Мир, 1985. — 509с.
4. *SAT Live* [Электронный ресурс]. — Режим доступа: www.satlive.org, свободный.

S.V. Listrovoy, A.V. Sidorenko

METHODS OF SOLUTION TO THE *k-SAT*-PROBLEM IS BASED ON ITS REDUCTION TO THE PROBLEM OF COVERING

An algorithm for solving the *k-SAT*-problem for the average polynomial time and 3-*SAT*-problem for the polynomial time. The proposed method can significantly reduce the time to solve *SAT*-problems.

K e y w o r d s: SAT-problem, polynomial reducibility.

REFERENCES

1. Hirsch, E.A. (2000), “New Worst-Case Upper Bounds for SAT”, *Journal of Automated Reasoning*, Vol. 24, no. 4, pp. 397-420.
2. Listrovoy, S.V. and Minuchin, S.V. (2012), “The method of solving the problems of the minimum vertex cover in an arbitrary graph and the problem of the lowest coverage”, *Elektronnoe modelirovanie*, Vol. 34, no. 1, pp. 29-43.
3. Papadimitriou, H. and Stayglits, K. (1985), *Kombinatornaya optimizatsiya. Algoritmy i slozhnost* [Combinatorial optimization. Algorithms and complexity], Mir, Moscow, Russia.
4. SAT Live, available at: http: // www.satlive.org.

Поступила 27.06.14
после доработки 03.07.15

ЛИСТРОВОЙ Сергей Владимирович, д-р техн. наук, профессор, профессор кафедры специализированных компьютерных систем Украинского государственного университета железнодорожного транспорта. В 1972 г. окончил Харьковское высшее военное командно-инженерное училище. Область научных исследований — задачи дискретной оптимизации и теории графов и их приложение к анализу вычислительных систем и сетей.

СИДОРЕНКО Андрей Владимирович, вед. инженер-программист Научного производственного предприятия «Стальэнерго» (г. Харьков). В 2001 г. окончил Харьковский военный университет. Область научных исследований — задачи дискретной оптимизации и теории графов и их приложения к анализу вычислительных систем и сетей.