

ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ В ДИСКРЕТНОЙ МАТЕМАТИКЕ

DOI 10.17223/20710410/2/24

УДК 519.7

МИНИМИЗАЦИЯ ФУНКЦИОНАЛОВ, АССОЦИИРОВАННЫХ С ЗАДАЧАМИ КРИПТОГРАФИЧЕСКОГО АНАЛИЗА АСИММЕТРИЧНЫХ ШИФРОВ

В.И. Дулькейт, Р.Т. Файзуллин, И.Г. Хныкин

Омский государственный университет им. Ф.М. Достоевского

E-mail: r.t.faizullin@mail.ru

В работе рассматриваются численные методы минимизации функционалов, ассоциированных с задачами криптографического анализа асимметричных шифров. Показано, что для задачи факторизации подход позволяет получить строго более чем 50 % бит, определяющих ключ.

Ключевые слова: криптографический анализ, алгоритм минимизации, ВЫПОЛНИМОСТЬ.

Концепция

Пусть $L(x)$ – пропозициональная формула в конъюнктивной нормальной форме на множестве булевых переменных $x \in B^N$, где $B = \{0, 1\}$. Задача ВЫПОЛНИМОСТЬ (SAT) заключается в том, чтобы найти решающий набор $x_0 \in B^N$, такой, что $L(x_0) = \text{ИСТИНА}$, или доказать, что решающего набора не существует.

Рассмотрим переход от задачи ВЫПОЛНИМОСТЬ к задаче поиска глобального минимума функционала вида (1).

Пусть дана КНФ:

$$L(x) = \bigcap_{i=1}^M c_i,$$

где c_i – дизъюнкты вида $c_i = \bigcup_{j < N} q_{i,j}(x_j)$. Здесь $q_{i,j}(x_j) = x_j$ или $\bar{x}_j$.

Сделаем переход к эквивалентной ДНФ:

$$L = \tilde{L}(x) = \bigcup_{i=1}^M \tilde{C}_i,$$

где $\tilde{C}_i$ – конъюнкты вида $\tilde{C}_i = \bigcap_{j < N} \tilde{Q}_{i,j}(x_j)$. Здесь $\tilde{Q}_{i,j}(x_j) = \bar{q}_{i,j}(x_j)$.

Рассмотрим функционал вида

$$\min_{x \in E^N} F(x) = \sum_{i=1}^M C_i(x), \text{ где } C_i(x) = \prod_{j=1}^N Q_{i,j}(x_j).$$

Здесь

$$Q_{i,j}(x_j) = \begin{cases} (1-x_j)^2, & \text{если } x_j \in C_i(x), \\ x_j^2, & \text{если } \bar{x}_j \in C_i(x), \\ 1, & \text{иначе.} \end{cases} \quad (1)$$

Суммирование ведется по всем M конъюнктам ДНФ, эквивалентной исходной КНФ. Соответствие между булевыми и вещественными переменными следующее: ЛОЖЬ $\rightarrow 0$, ИСТИНА $\rightarrow 1$.

Переход от булевой формулы к вещественной основан на использовании соответствия:

$$\begin{cases} y_i \vee y_j \rightarrow x_i + x_j, \\ y_i \wedge y_j \rightarrow x_i^2 x_j^2, \text{ где } \{y_i \in B, x_i \in R\}. \\ \bar{y}_i \rightarrow (1-x_i), \end{cases}$$

Легко заметить, что $\min_{x \in E^N} F(x) = 0$ соответствует достижению значения ИСТИНА на исходной КНФ.

Без потери общности можно рассмотреть 3-ДНФ, эквивалентную исходной КНФ:

$$J(x) = \sum_{\xi} z_i^2 z_j^2 z_k^2, \text{ где } z_i = \begin{cases} 1 - x_i, & \text{если } x_i \in c_i(x), \\ x_i, & \text{если } \bar{x}_i \in c_i(x), \end{cases} \quad c_i(x) - i \text{ триплет.} \quad (2)$$

Дифференцируя функционал по всем переменным x_i , получаем систему уравнений:

$$\sum_{\xi \in \Xi} z_j^2 z_k^2 x_i = \sum_{\xi \in \Lambda} z_j^2 z_k^2, \quad i = 1, 2, \dots, P, \quad (3)$$

где

$$\Xi = \{\xi, i \in \xi : x_i \in c_i(x)\}, \\ \Lambda = \{\xi, i \in \xi : \bar{x}_i \in c_i(x)\}$$

или

$$A_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \cdot x_i = B_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n), \quad i = 1, \dots, P.$$

Коэффициенты A_i и B_i связаны соотношением $A_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \geq B_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$.

Поясним выбор представления исходной КНФ именно в виде эквивалентной 3-ДНФ. Дифференцируя функционал $F(x)$ (см. (1)) по всем переменным x_i , получаем систему уравнений, аналогичную (3), но количество вкладов в A_i и B_i определяется длиной скобок. Любая процедура решения этой системы при произвольной длине скобок будет естественным образом приводить к большим ошибкам округления. Ограничивая число переменных в скобках, мы исключаем эту техническую трудность.

Рассмотрим систему (3) как нелинейное операторное уравнение:

$$\Phi(x) = 0. \quad (4)$$

Как показано в [5], применение метода Ньютона к решению данного уравнения неэффективно, так как решение принадлежит ядру производного оператора. Как альтернатива был предложен метод последовательных приближений с «инерцией»:

$$\left(\sum_{p=0}^K \sum_{\xi \in \Xi} \alpha_p x_i(t-p)^2 x_j(t-p)^2 \right) x_k(t+1) = \sum_{\xi \in \Lambda} x_j^2(t) x_k^2(t) \sim A \cdot x_k(t+1) = B, \quad (5)$$

$$\sum_{p=1}^K \alpha_p = 1, \quad \text{где } \alpha_p \in R[0,1].$$

Имеется в виду то, что итерации происходят для вещественных чисел, а итоговый или промежуточный вектор проектируется на B^N , и уже на булевом векторе проверяется SAT. Ниже мы опишем различные модификации и гибридизации метода последовательных приближений с «инерцией» в применении к решению задачи K-SAT и покажем способы повышения эффективности алгоритма.

Гибридизация алгоритма

Основная процедура состоит из последовательных итераций, которые совмещают метод последовательных приближений и сдвиг по антиградиенту, так как правая часть (3) – это не что иное, как антиградиент исходного функционала.

Итерация состоит из двух блоков. Первый блок определяется формулой (5), используется схема Зейделя. Суть схемы Зейделя в том, что при нахождении очередного $x_i(t+1)$ на $(t+1)$ -й итерации это значение подставляется вместо $x_i(t)$. Необходимо отметить, что реализация алгоритма допускает использование схемы Якоби, когда найденные $x_i(t+1)$ не используются в текущей итерации. Тесты показали, что схема Зейделя более устойчива в применении к решаемой задаче. Именно, после каждой итерации по схеме Зейделя значение функционала монотонно уменьшается, чего нельзя сказать о схеме Якоби. Кроме того, во всех случаях схема Зейделя быстрее приводила к решению. Отметим, что данное обстоятельство препятствует напрашивающейся простой схеме распараллеливания процесса решения с разделением данных.

Второй блок – реализация сдвига по градиенту. Рассмотрим (4). Пусть $\bar{x}(t)$ является решением, тогда $\Phi(\bar{x}(t)) = 0$. Уравнение (5) переписывается в виде $A(\bar{x}(t)) \cdot \bar{x}(t) - B(\bar{x}(t)) = 0$. Это необходимое условие, которому должен удовлетворять вектор решения. Если текущее t -е приближение $\bar{x}(t)$ не является решением, то $A_i(\bar{x}(t)) \cdot x_i(t) - B_i(\bar{x}(t)) = p_i \neq 0$. Для итеративной формулы: $A_i(\bar{x}(t)) \cdot x_i(t+1) - B_i(\bar{x}(t)) = p_i$. Следовательно, чтобы удовлетворить необходимому условию, необходимо перейти к вектору:

$$\bar{x}_i(t+1) = \bar{x}_i(t) + p_i / A_i. \quad (6)$$

Очевидно, что после реализации (6) возможна ситуация, когда $\bar{x}_i(t+1) \notin R[0,1]$. В этом случае необходимо штрафным способом ограничивать $\bar{x}_i(t+1)$, иначе метод начинает экспоненциально расходиться. Осо-

бенно это проявляется на K-SAT формулах при $K > 4$. При приближении к решению скорость сходимости может сильно уменьшаться, т.е. алгоритм формирует цикл, лежащий на некотором плато (поверхность, определяемая функционалом), и траектория, образованная последовательными приближениями, более не выходит за пределы этого плато. Чтобы сойти с плато и продолжить сходимость к решению, применяется т.н. метод смены траектории.

Метод смены траектории заключается в поиске нового вектора приближения, который бы обладал свойствами, не худшими, чем текущий вектор приближения, но позволял бы продолжить поиск решения. Суть метода нахождения такого вектора в следующем.

Рассмотрим 3-КНФ, эквивалентную исходной 3-ДНФ (1):

$$K(x) = \prod_{\xi} (z_i + z_j + z_k), \text{ где } z_i = \begin{cases} x_i, & \text{если } x_i \in c_i(x), \\ 1 - x_i, & \text{если } \bar{x}_i \in c_i(x), \end{cases} \quad c_i(x) - i \text{ триплет,} \quad (7)$$

$$K(x) = 1 \Leftrightarrow (z_i + z_j + z_k) \neq 0, \forall \xi.$$

Для данного приближения $\bar{x}$ рассмотрим множество переменных:

$$E0 = \{x_i \mid \exists \text{ триплет } c_i : x_i \text{ или } \bar{x}_i \in c_i \text{ \& } c_i(\bar{x}) = 0\}.$$

С вероятностью m_p поменяем значения x_i на противоположные. При этом вероятность того, что другие триплеты станут невыполнимыми, невысока. Экспериментально установлено, что полученный вектор x_0 обладает свойствами, не худшими, чем $\bar{x}$ (количество невыполнимых триплетов до и после операции примерно одинаково). Используя данный вектор x_0 в качестве нового начального приближения, алгоритм очень быстро (в большинстве случаев за 5 – 10 итераций) находит следующее приближение, на котором функционал $F(x)$ достигает значения не хуже, чем на векторе $\bar{x}$. При этом, очень часто, удается проскочить плато, но при дальнейшем движении по новой траектории метод может заикнуться на другом плато. Тогда метод смены траектории повторяется.

Дополнительно рассмотрим множество:

$$E1 = \{x_i \mid \exists \text{ триплет } c_i : x_i \text{ или } \bar{x}_i \in c_i \text{ \& } c_i(\bar{x}) = 1\}.$$

Введем вероятности m_p0 , m_p1 . С вероятностью m_p0 при смене траектории будем использовать множество $E0$. С вероятностью m_p1 – множество $E1$. Вероятность m_p0 влияет на величину изменения вектора и при этом количество невыполнимых триплетов не увеличивается. Вероятность m_p1 влияет на качественное изменение вектора, количество невыполнимых триплетов может увеличиться. В принципе, чем выше m_p1 , тем меньшую роль играют рестарты. Метод смены траектории применяется при достижении условия $|F(\bar{x}_2) - F(\bar{x}_1)| < \varepsilon_2$.

Преобразование исходной КНФ методом резолюции

Данное преобразование позволяет получить КНФ с меньшим количеством дизъюнктов и литералов, эквивалентную исходной. «Резольвента» – дизъюнкция конъюнктов, отличающихся знаком по единственной переменной. Все возможные резольвенты добавляются к КНФ и используются для вычисления других резольвент.

Дублирующие конъюнкты и тавтологии удаляются, и используется сокращенная процедура с глубиной рекурсии 1. Вычислительная сложность процедуры $O(n \cdot \log n)$. Метод резолюции в применении к КНФ, ассоциированных с задачами факторизации и дискретного логарифмирования (см. ниже), позволяет уменьшить исходное число конъюнктов до 50 % и иногда разрешить до 20 % переменных. Подробно о методе резолюций можно найти в [6]

Результаты численных экспериментов

После каждой модификации проводилось тестирование алгоритма для определения эффективности предложенных изменений. При тестировании использовалось несколько типов примеров: тесты с соревнованиями решателей SAT 2005 года www.lri.fr/~simon/, тесты библиотеки SATLib www.cs.ubc.ca/~hoos/, тесты, сформированные для задач факторизации и дискретного логарифмирования, тесты для КНФ больших размерностей, сформированные случайным образом. Подробные результаты представлены в [5].

Оказалось, что сдвиг по градиенту хорошо сокращает погрешности и ускоряет сходимость алгоритма. Вычислительные эксперименты со случайными формулами показали заметное уменьшение времени решения тестов. Число решенных примеров увеличилось примерно на 20 %. Применение данного приема позволило достаточно эффективно решать тесты uf20-91 (из 1000 тестов решены 703). Однако некоторые тесты решались только после задания определенного начального приближения, что говорит о необходимости рестартов. На примерах uf250-1065 алгоритм показал результат 6 % от стандартных трудных тестов (предыду-

щая версия алгоритма – 1 %). Тесты SAT-2005 (OKGenerator10000-42000 – 10000 переменных, 42000 скобок) использовались в сокращенной форме. Максимально удалось решить подформулы из 36000 скобок (предыдущая версия алгоритма – 35000 скобок). На подформулах из более чем 36000 скобок метод закидывается на некотором плато, что говорит о необходимости смены траектории.

Метод смены траектории существенно увеличил число решаемых примеров. Результаты представлены в табл. 1.

Таблица 1

Результаты тестирования алгоритма + метод смены траектории

Наименование теста	Количество литералов (N)	Количество дизъюнктов (M)	Число тестов	% решенных тестов	Максимальное число итераций
<i>Backbone-minimal Sub-instances (формулы с минимальным хребтом), 3-SAT</i>					
RTI	100	429	500	98,6	19988
BMS	100	<429	500	79,8	29831
<i>Controlled Backbone Size Instances (b – размер хребта), 3-SAT</i>					
CBS_b10	100	403	1000	100	38972
CBS_b10	100	449	1000	100	38880
CBS_b90	100	449	1000	98	29738
<i>Uniform Random 3-SAT (UF)</i>					
uf20-91	20	91	1000	100	448
uf250-1065	250	1065	100	98	9731
<i>SAT-encoded "Flat" Graph Colouring Problems</i>					
flat30-60	90	300	100	100	4317

Увеличение разрядности вычислений

Была исследована сходимость алгоритма при увеличении разрядности вычислений. Испытания с типами DOUBLE и FLOAT показали преимущество вычислений с двойной точностью. При переходе на тип DOUBLE количество решенных примеров увеличивается на 10 %, скорость сходимости в среднем также увеличивается. Дальнейшее увеличение разрядности к значимому эффекту не приводит.

Возможные практические применения

В последнее время наблюдается повышенный интерес к проблеме кодирования криптографических алгоритмов в терминах задачи выполнимости (SAT). В работе [4] эскизно иллюстрируется подход, позволяющий в принципе свести задачу факторизации к SAT. Целью данного исследования является построение алгоритмов генерации эквивалентных, но различных КНФ и последующей минимизации ассоциированного функционала для задач факторизации, дискретного логарифмирования и дискретного логарифмирования на эллиптической кривой.

Результаты работы алгоритма для задачи факторизации

Результаты приведены в табл. 2. Для группы задач длины до 72 бит факторизируемого числа были получены точные решения. При этом эффективность предложенного метода превосходит известные нам алгоритмы.

Таблица 2

Результаты тестирования полного алгоритма для задачи факторизации

Число бит в факторизируемом числе	Количество литералов	Количество дизъюнктов	Время решения методом последовательных приближений	Время решения алгоритмом RANOV (победитель 2005 г.)	Время решения алгоритмом SATz (один из лучших переборных алгоритмов)
20	254	4979	0,1 с	0,1 с	0,1 с
32	801	17867	2 мин	>1 ч	1,8 мин
40	990	22333	7 мин	>1 ч	12 мин
44	1199	27291	36 мин	>1 ч	>1 ч
48	1428	32741	3,5 ч	>10 ч	>10 ч
56	1946	45141	36 мин	>10 ч	>10 ч
60	2235	52079	10,2 ч	>20 ч	>20 ч
68	2873	67455	79 ч	>100 ч	>100 ч
72	3222	75881	168 ч	>200 ч	>200 ч

Другой интересный результат в том, что уже после первых нескольких сотен итераций метод находит более 59 % верных бит решения. Трудность заключается в распознавании, какие именно биты были опреде-

лены верно. При этом при росте числа бит исходного факторизуемого числа происходит рост процентного отношения числа верных бит для соотношения, определяющего факторизуемое число (рис. 1).

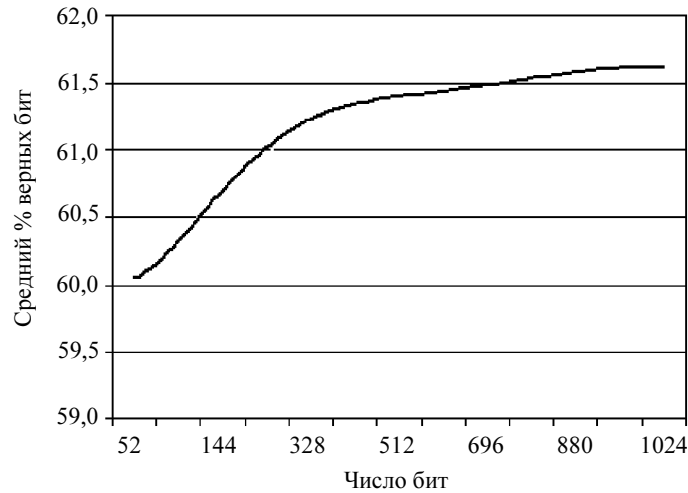


Рис. 1. Рост процентного отношения числа верных бит к числу бит факторизуемого числа

Для тестирования использовалось 50 различных тестов для каждой размерности. В качестве сомножителей выбирались числа, удовлетворяющие всем тестам, гарантирующим криптостойкость RSA. На рис. 1 представлены усредненные данные. Обратим внимание на то, что вне зависимости от размерности задачи для каждого теста проводилось всего 1000 итераций. Исходя из рисунка, можно сделать предположение о том, что рост числа верных бит в зависимости от длины факторизуемого числа имеет логарифмический характер.

Сведение к КНФ задачи факторизации

Рассмотрим непосредственно алгоритм сведения к КНФ задачи факторизации. Требуется для заданного числа n получить КНФ, решающий набор которой существует тогда и только тогда, когда n — составное число. Кроме того, решающий набор должен содержать все биты двоичного представления нетривиальных делителей n . Без потери общности рассмотрим классический алгоритм умножения «столбиком». Будем отождествлять биты сомножителей и результата с литералами (свободными логическими переменными). Результат умножения первого сомножителя на i -й бит второго можно представить в виде вектора $\bar{P}_i$. Именно суммирование всех этих векторов представляет основную сложность. Поэтому предлагается выполнять эту операцию последовательно с сохранением результата в промежуточных векторах $\bar{S}_k$.

Весь процесс вычисления можно разбить на три этапа относительно операции сложения:

1. Сложение векторов, составленных из произведений двух литералов. Выполняется один раз. В результате этой операции будет заполнен вспомогательный вектор сумм $\bar{S}_1$ и вычислено два младших бита результата. Условно данный этап можно записать так: $\bar{P}_1 + \bar{P}_2 = (\bar{S}_2, r_2, r_1)$.

2. Суммирование вектора $\bar{S}_k$ с вектором произведений. Выполняется $N - 3$ раз. В результате заполняется массив $\bar{S}_{k+1}$ и вычисляется очередной бит результата $\bar{S}_{i-2} + \bar{P}_i = (\bar{S}_{i-1}, r)$, $i = 3 \dots N - 1$.

3. Последнее суммирование вектора $\bar{S}_k$ с вектором произведений. Выполняется один раз. В результате вычисляются оставшиеся биты результата $\bar{S}_{N-2} + \bar{P}_N = (r_{2N} \dots r_N)$.

Теперь перейдем к рассмотрению идеи генерации КНФ. Простейший случай — приравнивание одного литерала другому: $x = y$. Данное равенство будет справедливо тогда и только тогда, когда истинна формула $(\bar{x} \vee y)(x \vee \bar{y})$.

Другим часто встречающимся выражением является

$$x = A \oplus B \oplus C. \quad (8)$$

Поступая аналогичным образом, получаем эквивалентную формулу:

$$(\bar{x} \vee (A \oplus B \oplus C))(\bar{x} \vee (\overline{A \oplus B \oplus C})) = (Ax \oplus Bx \oplus Cx \oplus \bar{x})(\bar{A}\bar{x} \oplus \bar{B}\bar{x} \oplus \bar{C}\bar{x}). \quad (9)$$

Для представления правой части в виде КНФ воспользуемся леммами 1 и 2.

Лемма 1.

$$\bigoplus_{i=1}^N x_i = \prod_{\{\delta_i\} \in M_N} (x_1^{\delta_1} \vee x_2^{\delta_2} \vee \dots \vee x_N^{\delta_N}),$$

где в левой части сумма по модулю 2, M_N – множество двоичных векторов длины N , содержащих чётное число нулей. Операция «возведения в степень» имеет стандартный для булевой алгебры смысл:

$$x^\delta = \begin{cases} \bar{x}, & \delta = 0, \\ x, & \delta = 1. \end{cases}$$

Лемма 2.

$$\bigvee_{i=1}^N x_i^{\delta_i} \vee \prod_{j=1}^L y_j^{\sigma_j} = \prod_{\{\pi_k\} \in 2^L / \{0,0,\dots,0\}} \left(\bigvee_{i=1}^N x_i^{\delta_i} \vee \bigvee_{j=1}^L (y_j^{\sigma_j})^{\pi_k} \right).$$

После применения леммы 1 будут получены конъюнкты следующего вида:

$$(d \vee \overline{abc} \vee x \overline{yc}),$$

т.е. можно выделить 3 вида дизъюнктов внутри каждого конъюкта:

1. Одиночные литералы (то, к чему следует стремиться: в правильной КНФ все дизъюнкты должны быть одиночными литералами).

2. Дизъюнкты вида $\prod x_i^{\delta_i}$, которые по правилу де Моргана можно легко свести к одиночным литералам.

3. Дизъюнкты вида $\prod x_i^{\delta_i}$, наиболее трудный случай, сведение скобок с такими дизъюнктами к КНФ иллюстрируется леммой 2.

Отдельного рассмотрения заслуживает операция вычисления переноса в следующий разряд при суммировании трёх слагаемых. Перенос может быть вычислен через соответствующую сумму:

$$c = \text{carry}(x, y, z, \text{sum}) = \overline{(\text{sum} \oplus xyz \oplus \overline{xyz})}.$$

Выполнение данного равенства эквивалентно истинности следующей формулы:

$$(\bar{c} \vee \overline{(\text{sum} \oplus xyz \oplus \overline{xyz})}) \cdot (c \vee (\text{sum} \oplus xyz \oplus \overline{xyz})).$$

Приведённое выражение можно преобразовать, положив

$$x = \bar{c}, A = \text{sum}, B = xyz, C = \overline{xyz}.$$

И далее описанной выше процедурой можно построить соответствующую КНФ. Трудоемкость полученного алгоритма оценивается как $O(n^2)$ в зависимости от количества бит исходного числа. Для факторизации числа, представляемого двоичным вектором длиной 1024 бит, получились КНФ с 500 000 переменными, 12 000 000 скобок. Отметим, что результат о превышении числа верных бит после нескольких тысяч итераций алгоритма (5) относится именно к числу переменных. Например, из 500 000 переменных мы получаем в среднем более чем 300 000 верных, что в силу очевидных соотношений между битами переноса по строке, отвечающей нулевому биту, может существенно облегчить решение основной задачи.

Сведение к КНФ других задач криптоанализа

Были получены аналогичные алгоритмы сведения для задач дискретного логарифмирования и дискретного логарифмирования на эллиптической кривой. Предложен алгоритм генерации множества эквивалентных КНФ для задачи факторизации, учитывающий неделимость на малые простые числа. Последнее обстоятельство позволяет строить параллельные версии приведенных выше алгоритмов и методикой «голосования бит» определять верные биты с определенной вероятностью.

Заключение

Разработанный метод не уступает известным методам решения SAT на многих группах тестовых примеров и превосходит их на тестах задачи факторизации больших размерностей. Показан рост относительного числа верно найденных бит для задачи факторизации с ростом размерности задачи. Так, для рабочего числа бит 1024 среднее значение верных бит равно 61,75 %, что больше стартового значения в 59,5 % для 50 бит. Кроме того, показано наличие «слабых» размерностей, для которых метод является эффективным. Анало-

гичные результаты получены и для задачи дискретного логарифмирования, но основным препятствием здесь является высокая размерность получающихся задач. Так, для 1024 бит число переменных в функционале оценивается величиной 1000 000 000, а число дизъюнктов на порядок больше. Результаты точного решения КНФ, эквивалентных задаче дискретного логарифмирования, приведены в табл. 3.

Таблица 3

Результаты тестирования полного алгоритма для задачи дискретного логарифмирования

Размерность, бит	Количество литералов	Количество дизъюнктов	Время решения методом последовательных приближений, с	Время решения алгоритмом RANOV, с (победитель 2005 г.)	Время решения алгоритмом SATz, с (один из лучших переборных алгоритмов)
18	28224	448018	63,57	97,23	81,16
20	38840	623239	108,20	>1800	>1800
22	51832	839032	182,73	>1800	>1800
24	67440	1099630	277,46	>1800	>1800
26	85904	1409250	417,71	>1800	>1800

Примечание. Знак «>» означает, что за указанное время решение найдено не было.

ЛИТЕРАТУРА

1. Беспалов Д.В. Пропозициональные алгоритмы в задачах несимметричной криптографии // Вестник ТГУ. Приложение. 2004. № 9(I).
2. Беспалов Д.В., Семёнов А.А. О логических выражениях для задачи 2-ФАКТОРИЗАЦИЯ // Вычислительные технологии. 2002. Т. 7. Ч. 2.
3. Найфе А. Методы возмущений. М.: Мир, 1976.
4. Файзуллин Р.Т. О решении нелинейных алгебраических систем гидравлики // Сибирский журнал индустриальной математики. 1999. № 2. С. 176 – 184.
5. Файзуллин Р.Т., Хныкин И.Г., Дулькейт В.И., Салаев Е.В. Алгоритм минимизации функционала, ассоциированного с задачей 3-SAT и его практические применения. Челябинск, 2007.
6. Хныкин И.Г. Модификация КНФ, эквивалентных задачам криптоанализа асимметричных шифров, методом резолюции // ИТМУ. 2007. № 8.
7. Cook S.A. The Complexity of Theorem Proving Procedures. Proceedings Third Annual ACM Symposium on Theory of Computing, May 1971.
8. Gu J., Purdom P.W., Franco J., Wah B.W. Algorithms for the Satisfiability Problem: A Survey // DIMACS Series in Discrete Mathematics and Theoretical Computer Science. 1996. P. 19 – 151.