2008 Вычислительные методы в дискретной математике № 2(2)
DOI 10.17223/20710410/2/25
УДК 519.7

АНАЛИЗ НЕКОТОРЫХ КРИПТОГРАФИЧЕСКИХ ПРИМИТИВОВ НА ВЫЧИСЛИТЕЛЬНЫХ КЛАСТЕРАХ[1]

А.А. Семенов, О.С. Заикин, Д.В. Беспалов, П.С. Буров, А.Е. Хмельнов

Институт динамики систем и теории управления СО РАН, г. Иркутск

E-mail: biclop@rambler.ru, oleg.zaikin@icc.ru, bespalov@altrixsoft.com, evle.zzz@gmail.com, alex@icc.ru

Работа является продолжением серии статей, посвященных обращению дискретных функций из класса, имеющего пересечения с различными областями кибернетики. В данный класс, в частности, попадают функции, используемые в современных криптосистемах в качестве шифрующих преобразований. В настоящей работе обсуждаются некоторые характерные моменты, касающиеся решения задач обращения рассматриваемых функций на многопроцессорных вычислительных системах.

Ключевые слова: *обращение дискретных функций*, *преобразования Цейтина*, *многопроцессорный кластер.*

Данная работа предполагает освещение имеющихся результатов и перспектив дальнейших исследований в решении булевых уравнений большой размерности с приложениями в задачах криптографии. Особое внимание уделяется использованию для решения рассматриваемых классов логических уравнений многопроцессорных вычислительных систем (кластеров).

На сегодняшний день основу программ, наиболее успешно работающих с логическими уравнениями, составляют алгоритмы решения SAT-задач. Тем самым один из доминирующих на данный момент подходов состоит в сведении исходной (разнородной) системы логических уравнений к уравнению вида

$$C(x_1, \ldots, x_n) = 1, \quad (1)$$

в левой части которого находится конъюнктивная нормальная форма (КНФ) над множеством булевых переменных $X = \{x_1, \ldots, x_n\}$.

В п. 1 работы предполагается дать краткий обзор техники преобразований Цейтина [1] – основного инструмента, позволяющего сводить системы логических уравнений к уравнениям в формате (1); п. 2 содержит краткое описание программного комплекса «Transalg», реализация в рамках которого преобразований Цейтина позволяет сводить к уравнениям вида (1) задачи обращения дискретных функций из широкого класса (данный класс включает в себя большинство функций, используемых в системах шифрования).

В п. 3 описывается технология крупноблочного распараллеливания SAT-задач для их последующего решения на вычислительных кластерах. Данная технология была с успехом применена в криптоанализе ряда генераторов ключевого потока [2, 3]. Здесь же отмечено, что используемая концепция крупноблочного параллелизма применима не только к SAT-задачам, но и к многочисленным классам задач с «булевыми данными».

В п. 4 подробно описан программный комплекс D-SAT – основной инструмент в решении задач обращения дискретных функций на вычислительных кластерах.

В п. 5 описывается технология построения прогнозов трудоемкости задач криптоанализа (в форме SAT-задач) при их решении на вычислительных кластерах. В основе данной техники лежит идеология крупноблочного параллелизма, позволяющая строить прогнозы параллельного времени решения SAT-задач при варьируемых значениях ряда входных параметров. В качестве иллюстрирующего примера рассматривается логический криптоанализ широко известной системы поточного шифрования A5/1.

1. Преобразования Цейтина логических уравнений

Преобразования Цейтина в применении к логическим уравнениям реализуют фундаментальную идею, которую можно назвать «концепцией подстановки». Данная идея, состоящая в замене некоторой части формулы новой переменной (с последующими дополнительными преобразованиями), возникает в различных областях математики. В контексте логических уравнений такого рода преобразования впервые были использованы Г.С. Цейтиным в 1968 г. в работе [1]. Данную статью следует считать, по-видимому, наиболее ранней работой, в которой анализировались вопросы сложности вывода в исчислении высказываний. В ней, в частности, было установлено, что совмещение стратегии общей резолюции (в смысле Дж.А. Робинсона, [4])

[1] Работа выполнена при поддержке гранта РФФИ № 07-01-00400-а и при частичной поддержке гранта Президента РФ НШ–1676.2008.1.

с преобразованиями Цейтина[1] позволяет на некоторых классах логических противоречий получить «экспоненциальный выигрыш» по сложности в сравнении с регулярной резолюцией. Статья [1] была перепечатана на английском языке в 1983 г. [6].

В несколько более общем (по сравнению с [1]) контексте преобразования Цейтина были использованы в [7] при формулировке основных концепций логического криптоанализа[2]. Примерно в то же время [8] было показано, что преобразования Цейтина обладают свойством консервативности в смысле [9] (см. также [10]). Этот факт очень важен по отношению именно к задаче обращения дискретных функций, поскольку он позволяет гарантировать ее корректное решение.

Напомним, что логическими уравнениями называются выражения вида

$$L(x_1, \ldots, x_n) = 0, L(x_1, \ldots, x_n) = 1,$$

где $L(x_1, \ldots, x_n)$ – формула исчисления высказываний над множеством булевых переменных $X = \{x_1, \ldots, x_n\}$. Поиск решения логического уравнения

$$L(x_1, \ldots, x_n) = \beta, \beta \in \{0,1\},$$

означает поиск такого набора $(\alpha_1, \ldots, \alpha_n)$, $\alpha_i \in \{0, 1\}$, $i \in \{1, \ldots, n\}$, что $L(x_1,...,x_n)|_{(\alpha_1,...,\alpha_n)} = \beta$. Здесь через $L(x_1,...,x_n)|_{(\alpha_1,...,\alpha_n)}$ обозначена подстановка в $L(x_1, \ldots, x_n)$ соответствующих значений переменных: $x_1 = \alpha_1$, …, $x_n = \alpha_n$. Если такого набора не существует, то рассматриваемое логическое уравнение не имеет решений.

Будем рассматривать задачу поиска решений логических уравнений в следующей общей постановке:

$$F\left(h_1\left(x_1^1,...,x_{r_1}^1\right),...,h_s\left(x_1^s,...,x_{r_s}^s\right)\right) = 1. \quad (2)$$

Здесь $h_1, \ldots, h_s$ – некоторые (в общем случае сложные) булевы функции. Все булевы функции здесь и далее полагаются всюду определенными. Положим

$$X = \{x_1,...,x_n\} = \bigcup_{i=1}^{s}\left\{x_1^i,...,x_{r_i}^i\right\},$$

таким образом, F: $\{0, 1\}^n \to \{0, 1\}$. Введем булеву функцию

$$g_{h_1}\left(x_1^1,...,x_{r_1}^1,u_1\right), g_{h_1} : \{0,1\}^{r_1+1} \to \{0,1\}:$$

$$g_{h_1}\left(x_1^1,...,x_{r_1}^1,u_1\right) = \begin{cases} 1, h_1\left(x_1^1,...,x_{r_1}^1\right) = u_1, \\ 0, h_1\left(x_1^1,...,x_{r_1}^1\right) \neq u_1. \end{cases}$$

Рассмотрим логическое уравнение

$$C\left(g_{h_1}\left(x_1^1,...,x_{r_1}^1,u_1\right)\right) \cdot F\left(u_1,...,h_s\left(x_1^s,...,x_{r_s}^s\right)\right) = 1. \quad (3)$$

Здесь $C\left(g_{h_1}\left(x_1^1,...,x_{r_1}^1,u_1\right)\right)$ – КНФ-представление булевой функции g_{h_1} над $\left\{x_1^1,...,x_{r_1}^1,u_1\right\}$. Переход от уравнения (2) к уравнению (3) – это одна итерация преобразований Цейтина.

Теорема 1 (см. [11]). Существует взаимно однозначное соответствие между множествами решений уравнений (2) и (3). Переход от произвольного решения уравнения (3) к соответствующему решению уравнения (2) осуществляется за линейное время.

Легко заметить, что, применив преобразования Цейтина к уравнению (2) полиномиально ограниченное от объема кодировки (2) число раз, можно перейти от уравнения (2) к уравнению вида (1), причем множество его решений будет равномощно множеству решений (2). Более того, вместо КНФ-представлений функций g можно использовать их представления в виде любых «нормальных форм».

2. Пропозициональное представление алгоритмов вычисления дискретных функций; программный комплекс «Transalg»

Преобразования Цейтина являются базой «пропозиционального подхода» к задаче обращения дискретных функций одного важного класса.

Под дискретными функциями здесь понимаются функции вида f_n: $\{0, 1\}^n \to \{0, 1\}^*$. Если натуральное семейство $\{f_n\}_{n \in N}$ вычисляется при помощи фиксированной программы M машины Тьюринга, то данное семейство функций называется алгоритмически вычислимым. Если при этом каждая функция f_n семейства определена всюду на $\{0, 1\}^n$, то говорят, что данное семейство образовано всюду определенными алгоритмически вычислимыми функциями. Для таких функций стандартным образом (см., например, [10]) вводится понятие алгоритмической сложности. Все возможные всюду определенные алгоритмически вычислимые за полиномиальное время дискретные функции образуют класс $\Im$. Очевидно, что данный класс включает в себя, в частности, все функции, используемые в криптосистемах в качестве шифрующих преобразований. За-

[1] Данный тип вывода получил название «расширенная резолюция» (см., например, [5]).

[2] Авторы на тот момент еще не знали, что фактически они используют преобразования Цейтина.

дача обращения произвольной функции $f_n \in \Im$ состоит в следующем: по произвольному $y \in \mathrm{Ran}\, f_n$ найти такой $x \in \mathrm{Dom}\, f_n = \{0, 1\}^n$, что $f_n(x) = y$ (через $\mathrm{Dom}\, f_n$ и $\mathrm{Ran}\, f_n$ обозначены соответственно область определения и область значений функции f_n).

В работах [7, 11, 2, 3] последовательно развивался подход к обращению функций класса $\Im$, базирующийся на идее пропозиционального представления прямых алгоритмов их вычисления. Данная концепция в своей основе восходит к работе С. Кука [12], однако ее практическое воплощение требует многочисленных детализаций.

Первая программная реализация «пропозиционального подхода» была осуществлена в [13]. Созданный программный комплекс (LC-комплекс) с успехом применялся в криптоанализе различных систем поточного шифрования. Основу LC-комплекса составлял LC-язык, представляющий собой фактически фрагмент языка С. Алгоритмы, записываемые на LC-языке, транслировались в логические выражения в формате КНФ при помощи специального транслятора (LC-транслятор). LC-комплекс оказался удобен для сведения задач криптоанализа блочных шифров к SAT-задачам. Однако для логического криптоанализа систем шифрования, в алгоритмах которых используются условные переходы, возможностей LC-комплекса было недостаточно. На сегодняшний день данный комплекс не удовлетворяет многообразию задач, вписывающихся в концепцию пропозиционального подхода к обращению функций класса $\Im$.

Все перечисленные причины привели к выводу об актуальности создания нового программного комплекса для пропозиционального представления алгоритмов. Данный проект получил название «Transalg» [14]. В настоящий момент проект находится в стадии активной разработки. Комплекс «Transalg» имеет модульную архитектуру и представляет собой среду, в которую интегрированы компоненты, осуществляющие трансляцию алгоритмов, преобразования форматов, решение логических уравнений и проверку получаемых решений. Модульность архитектуры Transalg'а позволяет задействовать для решения логических уравнений различные методы (SAT-подход, BDD-подход, работа с системами полиномиальных уравнений над GF(2) и др.).

3. Крупноблочный параллелизм в булевых задачах; прогнозирование параметров распараллеливания

Кратко опишем технологию крупноблочного распараллеливания, использованную в решении SAT-задач и апробированную на задачах логического криптоанализа некоторых генераторов двоичных последовательностей [2, 3]. В [15] показано, что основные идеи [2] переносятся без существенного изменения на разнообразные задачи с «булевыми данными». В изложении данного материала в основном будем следовать работе [15].

Отметим, что различные задачи, использующие двоичные данные, можно объединить в контексте общей постановки «удовлетворения ограничениям». А именно, будем рассматривать задачи следующего типа. Дан набор формул-условий вида

$$\Phi(x_{i_1^1},...,x_{i_{r_1}^1}) \rhd A_1,...,\Phi(x_{i_1^m},...,x_{i_{r_m}^m}) \rhd A_m, \tag{4}$$

где $\bigcup_{j=1}^{m}\{x_{i_1^j},...,x_{i_{r_j}^j}\} = \{x_1,...,x_n\} = X$ – множество переменных, принимающих значения в $\{0, 1\}$, $\rhd \in \{=,\geq\}$, $A_j, j \in \{1, \ldots, m\}$ – элементы GF(2) или целые числа. Формулы (4) – это выражения вида «КНФ = 1», системы линейных неравенств вида

$$A \cdot x \geq b,$$

(где A – матрица размерности $m \times n$ с элементами из множества $\{-1, 0, 1\} \cap Z$; b – целочисленный вектор размерности m), либо полиномиальные уравнения над полем GF(2). Требуется найти вектор значений истинности переменных из X, удовлетворяющий всем условиям (4). Такой вектор будем называть решением системы (4).

Для произвольного множества $X^d = \{x_{i_1},...,x_{i_d}\}$, $\{i_1, \ldots, i_d\} \subseteq \{1, \ldots, n\}$, определим подстановку набора значений переменных

$$x_{i_1} = \alpha_{i_1},...,x_{i_d} = \alpha_{i_d}, \alpha_{i_j} \in \{0,1\}, j \in \{1,...,d\}, \tag{5}$$

в произвольную формулу вида (4) как замену вхождения переменной $x_{i_j}, j \in \{1,...,d\}$, константой α_{i_j} с последующим выполнением соответствующих преобразований (когда это возможно). Преобразования, являющиеся результатами применения подстановок типа (5) к выражениям вида (4), – это вычеркивания соответствующих литералов и дизъюнктов из КНФ, перенос целых чисел (с противоположным знаком) в правую часть линейных неравенств, а также вычеркивание мономов из булевых полиномов либо замена некоторых мономов на константу «1» с последующим сложением по mod 2 нескольких констант. Обозначим через 2^{X^d} множество, образованное всевозможными двоичными векторами длины d, каждый из которых определяет

значения истинности переменных из X^d. Каждому вектору $\alpha = (\alpha_1, ..., \alpha_d) \in 2^{X^d}$ поставим в соответствие результат следующей подстановки в выражения (4): $x_{i_1} = \alpha_1, ..., x_{i_d} = \alpha_d$. Тем самым вектору $\alpha = (\alpha_1, ..., \alpha_d) \in 2^{X^d}$ сопоставляется набор преобразованных выражений типа (4), который будем обозначать через Φ^α. Таким образом, множество X^d задает естественное соответствие между набором формул вида (4) и семейством формул $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$. Каждая формула данного семейства есть либо логическое уравнение вида «КНФ = 1», либо система линейных неравенств вида (2), либо система полиномиальных уравнений над полем GF(2). Будем также говорить, что множество X^d задает декомпозицию задачи решения системы (4) на семейство задач поиска решений систем $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$. Дополнительно полагаем, что при $d = 0$ результатом такой декомпозиции является исходная система типа (4).

Несложно видеть, что если двоичный вектор β – решение системы Φ^α при фиксированном $\alpha \in 2^{X^d}$, то компоненты векторов α и β дают компоненты некоторого решения исходной системы (4). Наоборот, если γ – произвольное решение (4), то часть его компонент определяет некоторый вектор $\alpha \in 2^{X^d}$, а оставшиеся компоненты определяют вектор β, являющийся решением системы Φ^α. Таким образом, задача поиска решения системы (4) сводится к задаче поиска решений систем Φ^α, $\alpha \in 2^{X^d}$, причем (4) не имеет решений (несовместна) тогда и только тогда, когда при всех $\alpha \in 2^{X^d}$ системы Φ^α несовместны.

Задачи поиска решений систем вида Φ^α – это, по сути, частные случаи исходной задачи, имеющие меньшую размерность, и для их решения можно использовать многопроцессорный вычислительный кластер. Более точно, упорядоченное некоторым образом семейство систем $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$ может рассматриваться как очередь заданий для *r*-процессорного кластера. Именно такой подход был использован в [2] применительно к решению SAT-задач.

В этой же работе была предложена процедура статистического прогнозирования размерности множества X^d, то есть значения параметра *d*, при котором суммарное время решения исходной задачи на вычислительном кластере минимально. Здесь мы переносим данный результат работы [2] на общую задачу поиска решений систем вида (4).

Предположим, что рассматривается некоторая задача типа (4). Через *S* обозначаем решатель, используемый для поиска ее решений. Решатель – это некоторая программа, помимо базового алгоритма использующая разнообразные эвристики, в том числе и эвристики удаления нерелевантных ограничений. Термин «прогнозная функция» обозначает семейство функций, параметрически зависящих от решателя *S*. Аргументами данных функций являются случайные выборки систем из множеств $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$. Прогнозная функция является частично определенной на заданном множестве выборок и каждой «точке» своей области определения ставит в соответствие некоторое положительное рациональное число. Знание глобального минимума данной функции на ее области определения позволяет получить представление (прогноз) относительно минимального общего объема вычислений, требуемого для решения распараллеленной исходной задачи.

Случайные выборки из множеств $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$ имеют смысл лишь в тех случаях, когда мощность данного множества (величина 2^d) слишком велика для его полного перебора. Поэтому при определении прогнозной функции имеет смысл рассматривать две ситуации: когда число 2^d больше, чем некоторая положительная константа ρ_0, и когда 2^d не превосходит ρ_0. За ρ_0 можно принять, например, число, сопоставимое с числом процессоров в кластере.

Далее полагаем, что все случайные выборки имеют фиксированный объем *q*. Каждому значению параметра $d \in \{0, 1, \ldots, n\}$, такому, что $2^d > \rho_0$, ставится в соответствие множество векторов $\{\alpha^1, \ldots, \alpha^q\}$, выбираемых из 2^{X^d} в соответствии с заданным на нем равномерным распределением, а также множество (выборка) систем $\Theta_d = \{\Phi^{\alpha_1}, ..., \Phi^{\alpha_q}\}$. Каждому значению параметра $d \in \{0, 1, \ldots, n\}$, такому, что $2^d \le \rho_0$, ставится в соответствие множество 2^{X^d} и множество систем $\{\Phi^\alpha\}_{\alpha \in 2^{X^d}}$.

Фиксируем некоторый решатель *S*. Обозначим через $t(\Phi')$ время работы (число битовых операций) решателя *S* на произвольном входе $\Phi' \in \Theta_d$. Введем в рассмотрение функцию

$$\tau_S(\Theta_d) = \sum_{\Phi' \in \Theta_d} t(\Phi').$$

Значением данной функции при каждом фиксированном $d \in \{0, 1, \ldots, n\}$ является суммарное время (число битовых операций) работы решателя *S* по всем системам из Θ_d.

Следует учитывать, что при некоторых значениях параметра d (например, при $d = 0$) системы Θ_d могут оказаться очень сложными для решателя S, и в этом случае время подсчета соответствующего значения прогнозной функции может превысить разумные границы. Для учета данного факта в рассмотрение вводится специальная функция $g(\Phi) = p(|\Phi|)$, где $p(\cdot)$ – некоторый полином, а через $|\Phi|$ обозначен объем двоичной кодировки исходной системы вида (4).

Допустим, что в соответствии с перечисленными правилами построено семейство выборок $\Theta = \{\Theta_d\}_{d\in\{0,1,\ldots,n\}}$ (при фиксированном ρ_0). Прогнозную функцию определим следующим образом:

$$T(\Theta_d) = \begin{cases} \dfrac{2^d}{q}\cdot\tau_S(\Theta_d),\ 2^d > \rho_0,\ \tau_S(\Theta_d) < g(|\Phi|), \\ \tau_S(\Theta_d),\ 2^d \le R_0,\ \tau_S(\Theta_d) < g(|\Phi|), \\ \infty,\ \tau_S(\Theta_d) \ge g(|\Phi|). \end{cases} \quad (6)$$

Запись «$T(\Theta_d) = \infty$» означает, что функция не определена на выборке Θ_d. Рациональное число $T(\Theta_d)$ является прогнозом общего объема битовых операций, требуемого для решения исходной SAT-задачи при декомпозиции исходной системы вида (4) на семейство систем $\{\Phi^\alpha\}_{\alpha\in 2^{X^d}}$. Тем самым задача прогнозного планирования оптимального по трудоемкости параллельного вычисления сводится к задаче минимизации функции T на множестве Dom $T \subseteq \Theta$. Знание глобального минимума функции T на Dom T дает представление о возможности параллельного решения SAT-задачи для КНФ C «за разумное время».

Далее приведен основной результат работы [15], который является непосредственным обобщением теоремы, доказанной в [2] в отношении SAT-задач.

Теорема 2 (см. [15]). Пусть Φ – произвольная булева система вида (4), $\Theta=\{\Theta_d\}_{d\in\{0,\,1,\,\ldots,\,n\}}$ – произвольное семейство выборок систем из множества $\{\Phi^\alpha\}_{\alpha\in 2^{X^d}}$. Для произвольной прогнозной функции T: $\Theta \to Q$ вида (6) справедливы следующие свойства:

1. Dom $T \neq \varnothing$.
2. Глобальный минимум T на Dom T находится в общем случае за время, ограниченное полиномом от $|\Phi|$.

В работе [15] было продемонстрировано, что технология крупноблочного параллелизма, использующая прогнозные функции, позволяет добиваться на задачах криптоанализа некоторых генераторов (в частности, порогового генератора) «сверхлинейного» ускорения. Как показано в [15], данный феномен является прямым следствием неполноты используемого решателя.

4. Пакет D-SAT

Для реализации описанной выше технологии крупноблочного распараллеливания применительно к SAT-задачам был разработан пакет прикладных программ Distributed-SAT (D-SAT) [16]. Пакет функционирует на вычислительном кластере под управлением инструментального комплекса DISCOMP [3, 17]. Описание данного программного комплекса здесь приводится.

Библиотека программ пакета D-SAT включает модуль расщепления, модуль SAT-решателя, модуль прогнозирования, модуль-анализатор и транспортный модуль. Модули пакета реализованы на языке C++, причем все они являются платформонезависимыми.

Основные входные данные модуля расщепления – это файл с исходной КНФ в формате DIMACS, диапазон значений параметра d (левая и правая границы) и фиксированное значение q, определяющее число КНФ в произвольной случайной выборке. Обозначим через d_* и d^* натуральные числа, определяющие соответственно верхнюю и нижнюю границы интервала, в котором изменяются значения d. Для каждого значения диапазона параметров декомпозиции строится отдельное семейство КНФ. В случае $q < 2^d$ из декомпозиционного семейства случайным образом выбираются q КНФ. Если $q = 0$ или $q \ge 2^d$, то выборкой является все рассматриваемое семейство. Результатом декомпозиции исходной КНФ является набор файлов с КНФ декомпозиционного семейства, представленными в формате DIMACS. Этот набор в процессе работы комплекса DISCOMP описывается в виде параллельного списка.

Вычислительное ядро модуля SAT-решателя составляет программа minisat [18], вообще говоря, могут использоваться различные версии данной программы. По входному файлу, в котором КНФ представлена в формате DIMACS, модуль SAT-решателя осуществляет поиск выполняющего данную КНФ набора либо выдает значение *NULL*, если таковой отсутствует. В выходной файл каждой копии модуля SAT-решателя записывается информация о результатах его работы. Если в процессе обработки параллельного списка для какой-либо КНФ найден выполняющий набор, то все вычисления прекращаются и запускается модуль-анализатор.

Модуль прогнозирования обрабатывает файлы с результатами работы модуля SAT-решателя и производит их статистический анализ, на основе которого строится прогноз оптимальных параметров декомпозиции. При этом дополнительно учитываются все транспортные расходы.

Аналитический модуль проверяет найденный выполняющий набор на корректность, вычисляет среднее время решения SAT-задач для списка КНФ, минимальное и максимальное время, а также сравнивает прогнозируемое время решения с реальным временем. Решение SAT-задачи и результаты анализа записываются в итоговый файл.

Процесс решения SAT-задачи состоит из двух этапов: а) прогнозирование наиболее оптимальных (с точки зрения вычислительных затрат) параметров декомпозиции; б) решение всех SAT-задач из декомпозиционного семейства, определяемого найденными на этапе прогнозирования параметрами декомпозиции.

На этапе прогнозирования выполняются следующие шаги (рис. 1):

1. На вход модулю расщепления подается файл с исходной КНФ в формате DIMACS, значения d_*, d^* и q. Модуль расщепления для каждого значения d строит соответствующую выборку КНФ с учетом значения q. Полученные файлы КНФ в формате DIMACS объединяются в параллельный список и передаются на модуль SAT-решателя.

2. Для каждой КНФ данного списка решается соответствующая SAT-задача – доказывается невыполнимость либо находится выполняющий набор.

3. После того как решены SAT-задачи для всех КНФ из списка, при помощи модуля прогнозирования производится статистический анализ полученных результатов и на его основе вычисляется d_0 – прогнозное значение оптимального параметра декомпозиции.

В процессе прогнозирования существует возможность того, что будет найден набор, выполняющий исходную КНФ. В этом случае решение задачи считается найденным и вычислительный процесс завершается.

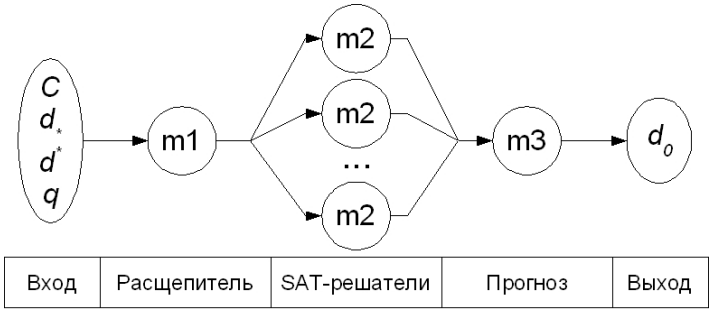

Рис. 1. Схема взаимодействия модулей пакета D-SAT на этапе прогнозирования

На этапе решения задачи выполняются следующие шаги (рис. 2):

1. На вход модулю расщепления подается файл с исходной КНФ в формате DIMACS, полагается $d_* = d^* = d_0$, где значение d_0 найдено на этапе прогнозирования, а также $q = 0$. Модуль расщепления строит семейство файлов КНФ C_1, …, C_k. Полученные КНФ объединяются в параллельный список и передаются на модуль SAT-решателя.

2. Для каждой КНФ полученного списка решается соответствующая SAT-задача – доказывается невыполнимость либо находится выполняющий набор.

3. Запускается модуль-анализатор.

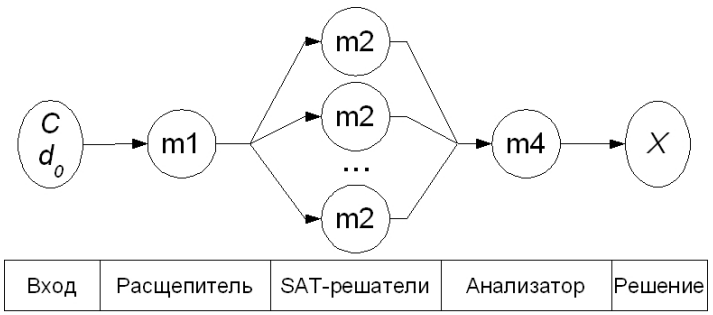

Рис. 2. Схема взаимодействия модулей пакета D-SAT на этапе решения

Обмен данными между вычислительными узлами кластера осуществляет специальный транспортный модуль пакета D-SAT, основной задачей которого является рассылка КНФ из параллельного списка. Каждая КНФ представляется в виде файла в формате DIMACS, состоящего из трех частей:

1) заголовок вида «*p cnf* [*число переменных*] [*число дизъюнктов*]»;

2) список из d однолитеральных дизъюнктов, построенных по соответствующим значениям переменных декомпозиции;

3) список дизъюнктов исходной КНФ.

В таком представлении у всех КНФ декомпозиционного семейства первая и третья части одинаковы, а различаются лишь вторые части. Назовем файлом различий файл со списком однолитеральных дизъюнктов произвольной КНФ семейства.

Работа транспортного модуля включает этап препроцессинга, в ходе которого на каждый вычислительный узел передается файл с исходной КНФ. В дальнейшем на вычислительные узлы передаются только файлы различий КНФ. Каждая отдельная SAT-задача формируется на основе информации из файла различия и файла с исходной КНФ. Описанная схема позволяет значительно сократить транспортные расходы.

5. Формирование декомпозиционных множеств; оптимизация параллельных планов в задачах логического криптоанализа; прогноз параллельного логического криптоанализа генератора A5/1

Довольно очевидный факт состоит в том, что значительное влияние на эффективность параллельного решения SAT-задачи может оказывать выбор декомпозиционного множества X^d. Здесь мы приводим некоторые общие схемы построения декомпозиционных множеств, дающих оптимальные по вычислительным затратам параллельные планы[1]. В качестве иллюстрации описываемого подхода рассматривается задача логического криптоанализа широко известной системы потокового шифрования A5/1 (рис. 3).

Далее мы рассматриваем задачу криптоанализа только генератора ключевого потока в данном шифре, оставляя за скобками собственно протокол. Описание генератора взято из [19]. В соответствии с этим описанием, в генераторе A5/1 используются 3 регистра сдвига с линейной обратной связью (РСЛОС, LFSR), задаваемые следующими полиномами обратной связи: РСЛОС 1: $X^{18} + X^{17} + X^{16} + X^{13} + 1$ (длина регистра 19 бит); РСЛОС 2: $X^{21} + X^{20} + 1$ (длина регистра 22 бита); РСЛОС 3: $X^{22} + X^{21} + X^{20} + X^7 + 1$ (длина регистра 23 бита).

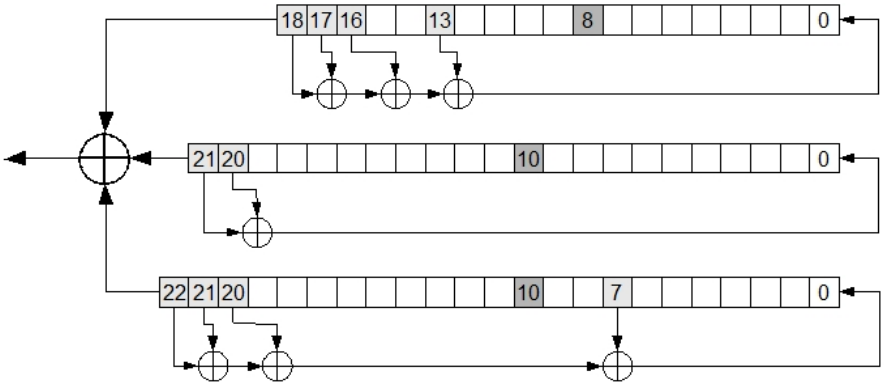

Рис. 3. Схема работы генератора ключевого потока шифра A5/1

В каждом такте могут сдвигаться не все регистры. Сдвиг регистра с номером m, $m \in \{1, 2, 3\}$, происходит, если значение функции $\chi_m\left(b_s^1, b_s^2, b_s^3\right)$ равно 1, и не происходит, если значение данной функции равно 0. Через b_s^1, b_s^2, b_s^3 здесь обозначены значения т.н. «серединных битов» текущего шага, то есть битов, находящихся в данный момент в девятой ячейке первого РСЛОС, и в ячейках с номером «11» РСЛОС2 и РСЛОС3 (данные ячейки на рис. 3 отмечены более темной заливкой, нумерация ячеек ведется справа налево). Функция $\chi_m(\cdot)$ определяется следующим образом:

$$\chi_m\left(b_s^1, b_s^2, b_s^3\right) = \begin{cases} 1, b_s^m = \text{majority}\left(b_s^1, b_s^2, b_s^3\right), \\ 0, b_s^m \neq \text{majority}\left(b_s^1, b_s^2, b_s^3\right), \end{cases}$$

где $\text{majority}(x, y, z) = x \cdot y \vee x \cdot z \vee y \cdot z$ (функция большинства).

Проблема построения оптимальной декомпозиции для параллельного решения задачи криптоанализа A5/1 оказалась весьма нетривиальной. Были исследованы различные схемы построения декомпозиционных множеств. Первые стратегии основывались на идеях, ранее успешно примененных в параллельном криптоа-

[1] Термин «оптимальность» здесь, конечно же, понимается в отношении конкретного SAT-решателя.

нализе ряда генераторов (пороговый, суммирующий, Гиффорда [3]). В частности, использовалась схема подстановки подряд идущих битов (в естественной нумерации) инициализирующей последовательности. Тем самым, например, при $d = 16$ в декомпозиционное множество включались переменные, кодирующие начальные состояния первых 16 ячеек РСЛОС1. При этом, исходя из некоторых «разумных» предположений, сначала строилось базовое декомпозиционное множество, а затем предпринимались попытки его усечения на основе значений соответствующих прогнозных функций. Ниже представлена схема (рис. 4) построения базового декомпозиционного множества X^d, состоящего из 31 переменной.

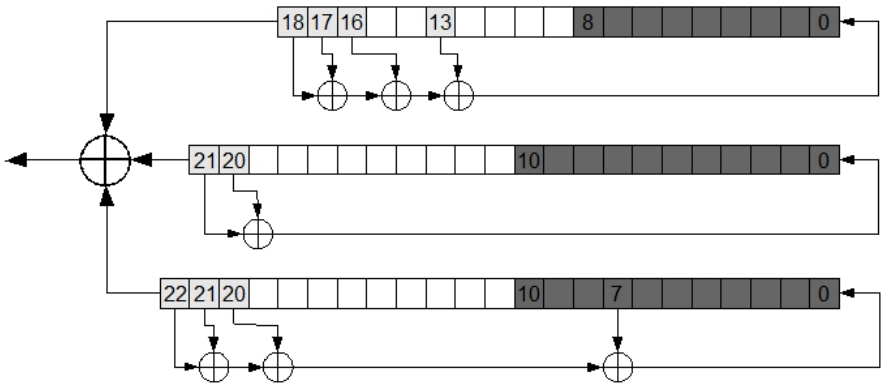

Рис. 4. Схема построения декомпозиционного множества из 31 переменной (X^d)

В соответствии с данной схемой предлагается включить в X^d переменные, кодирующие начальные состояния ячеек регистров, начиная с первых ячеек до ячеек, содержащих серединные биты (соответствующие ячейки на рис. 4 отображены темной заливкой). Иными словами, имеем декомпозиционное множество

$$X^d = \{x_1, \ldots, x_9, x_{20}, \ldots, x_{30}, x_{42}, \ldots, x_{52}\}. \quad (7)$$

Данный выбор продиктован следующими естественными соображениями. Произвольная фиксация битов из построенного таким образом множества X^d индуцирует точные значения серединных битов для значительного числа последующих состояний всех трех регистров. Но серединные биты являются наиболее информативными битами, поскольку именно они определяют, какая ветвь соответствующего условия (значение функции большинства) имеет место.

Довольно неожиданным оказался тот факт, что построенное в соответствии с (7) множество X^d не удалось существенно уменьшить за счет технологии прогнозных функций. В проводимых статистических экспериментах для каждого варианта декомпозиционного множества и начального фрагмента ключевого потока некоторой фиксированной длины строились случайные выборки объемом 1000 КНФ. Для каждой такой выборки считалось значение прогнозной функции – среднее значение времени решения SAT-задач по выборке, умноженное на мощность пространства перебора (см. п. 3). Во всех вычислительных экспериментах SAT-задачи решались SAT-решателем minisat версии 1.2. Каждая задача решалась на одном ядре процессора Intel E8400.

Далее в табл. 1 приведены значения среднего времени решения SAT-задач для спектра различных длин выхода и для различных вариантов формирования декомпозиционных множеств. Все эти варианты тем не менее в концептуальном смысле были близки к (7). Например, декомпозиционное множество, состоящее из 28 переменных, формировалось в соответствии со схемой на рис. 5 (в контексте оговоренных выше обозначений).

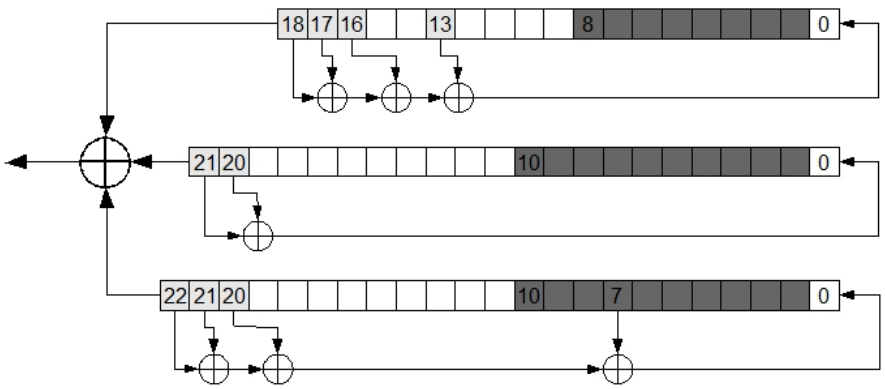

Рис. 5. Схема построения декомпозиционного множества из 28 переменных

Наилучшие итоговые результаты дало декомпозиционное множество, обозначаемое далее через X_*^d:

$$X_*^d = \{x_1,...,x_9,x_{21},...,x_{30},x_{42},...,x_{52}\}. \quad (8)$$

Таким образом, данное множество формировалось по следующей схеме (рис. 6):

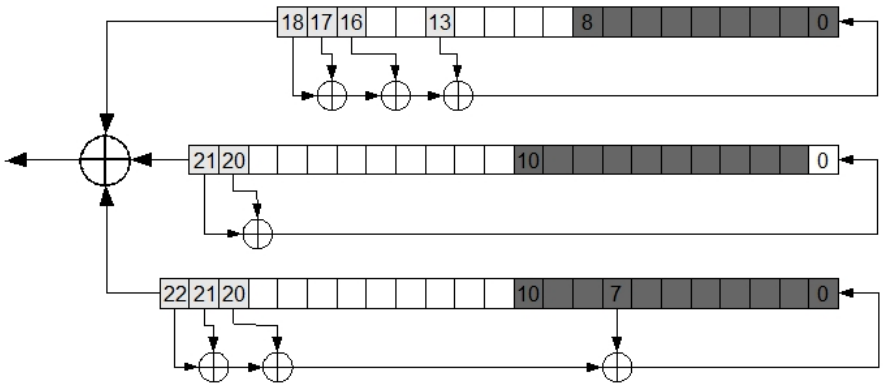

Рис. 6. Схема построения декомпозиционного множества из 30 переменных (X_*^d)

Таблица 1

Среднее время решения SAT-задач на одном ядре процессора Intel E8400 при различных вариантах декомпозиции и разных длинах выхода (объем каждой случайной выборки – 1000 КНФ, решатель minisat 1.2)

Число переменных	Длина выхода				
	144	160	176	192	208
28	4,241943	4,03173	4,79874	5,492782	5,1268
29	2,041447	2,062648	2,068609	1,888878	2,139253
30	0,848397	0,849213	0,948331	0,870502	1,006667
31	0,423078	0,428842	0,471429	0,420562	0,515448
32	0,229554	0,252899	0,320812	0,253239	0,267752
33	0,147385	0,151029	0,161326	0,156497	0,171918

Прогноз эффективности решения параллельной задачи криптоанализа требует также учета времени, затрачиваемого на пересылку файлов к процессорам/ядрам кластера (транспортные расходы). Благодаря схеме передачи данных, используемой в D-SAT (см. п. 4), эти затраты крайне незначительны – напомним, что «формулировкой» каждой новой задачи, передаваемой от центрального процессора на произвольный процессор/ядро, является двоичный вектор, длина которого равна мощности декомпозиционного множества. В табл. 2 приведены суммарные транспортные расходы на пересылку векторов длин 28 – 33 для двух различных типов соединений между вычислительными узлами (Gigabit Ethernet и Infiniband), наиболее часто используемых в современных кластерах. Данный прогноз был осуществлен на кластере Blackford ИДСТУ СО РАН [20] относительно системы Gigabit Ethernet и экстраполирован на систему Infiniband.

Таблица 2

Транспортные расходы при различных типах соединений между вычислительными узлами

Количество пересылаемых файлов	Транспортные расходы с интерконнектом Gigabit Ethernet, ч	Транспортные расходы с интерконнектом Infiniband, ч
2^{28}	1,5	0,75
2^{29}	3	1.5
2^{30}	6	3
2^{31}	12	6
2^{32}	24	12
2^{33}	48	24

Как уже отмечалось выше, все численные эксперименты проводились на процессоре Intel E8400. В табл. 3 приведены сравнительные характеристики данного процессора и процессора Intel E5472.

Кластер Т-60 [21] на момент написания данной статьи занимает 1-е место в списке 50-ти самых мощных суперкомпьютеров в СНГ [22]. Т-60 состоит из 1250 четырехъядерных процессоров E5472.

Таблица 3

Характеристики процессоров

Название процессора	Intel E8400	Intel E5472
Число ядер	2	4
Частота ядра	3,0 ГГц	3,0 ГГц
Частота шины	1333 МГц	1600 МГц
Технология производства	45 нм	45 нм
Кэш L2	6 Мб	12 Мб

Из данной таблицы видно, что мощность одного ядра процессора Intel E8400 сопоставима с мощностью одного ядра процессора Intel E5472 (незначительное отличие лишь в частоте шины). Исходя из этого факта, мы сочли возможным напрямую экстраполировать все приведенные выше результаты численных экспериментов применительно к процессу решения задачи логического криптоанализа генератора A5/1 на кластере Т-60. Наилучшие результаты приведены ниже.

1. Для декомпозиционного множества X^d_*, построенного в соответствии с (8), и 144 бит выходной последовательности прогноз времени решения задачи криптоанализа A5/1 (нахождение инициализирующей последовательности) на Т-60: 54 часа при интерконнекте Infiniband, 57 часов при интерконнекте Gigabit Ethernet.

2. Для декомпозиционного множества X^d, построенного в соответствии с (7), и 192 бит выходной последовательности прогноз времени решения задачи криптоанализа A5/1 на Т-60: 56 часов при интерконнекте Infiniband, 62 часа при интерконнекте Gigabit Ethernet.

Заключение

Описанные в данной работе подходы к обращению дискретных функций на многопроцессорных системах представляются весьма перспективными ввиду впечатляющего роста производительности вычислительной техники. Основной причиной этого, конечно же, следует считать многоядерные технологии, динамичное развитие которых делает возможным построение кластеров с гигантскими вычислительными мощностями (Т-60 [21], СКИФ Cyberia, [23]). Рассматриваемые в настоящей работе задачи и подходы к их решению интересны своей высокой масштабируемостью (прямая зависимость эффективности решения от числа процессоров/ядер).

В качестве дальнейших перспектив данного направления следует отметить удачно вписывающуюся в его контекст проблему анализа криптографических хэш-функций. Позитивной особенностью SAT-задач, кодирующих проблемы обращения хэш-функций, является тот факт, что соответствующие КНФ выполнимы более чем на одном наборе (как правило, число выполняющих наборов есть некоторая экспонента от длины входа). Это делает возможным использование новых алгоритмических подходов для решения такого рода SAT-задач (например, алгоритмы, базирующиеся на идеологии локального поиска).

ЛИТЕРАТУРА

1. *Цейтин Г.С.* О сложности вывода в исчислении высказываний // Зап. научн. семинаров ЛОМИ АН СССР. 1968. Т. 8. С. 234 – 259.
2. *Заикин О.С.*, *Семенов А.А.* Технология крупноблочного параллелизма в SAT-задачах // Проблемы управления. 2008. № 1. С. 43 – 50.
3. *Заикин О.С.*, *Семенов А.А.*, *Сидоров И.А.*, *Феоктистов А.Г.* Параллельная технология решения SAT-задач с применением пакета прикладных программ D-SAT // Вестник ТГУ. Приложение. 2007. № 23. С. 83 – 95.
4. *Robinson J.A.* A machine-oriented logic based on the resolution principle // J. ACM. 1965. V. 12. No. 1. P. 23 – 41. [Перевод: *Робинсон Дж.А*. Машинно-ориентированная логика, основанная на принципе резолюций // Кибернетический сборник. Новая серия. М.: Мир, 1970. Вып. 7. С. 194 – 218.]
5. *Haken A.* The intractability of resolution // Theoretical Computer Science. 1985. V. 39. P. 297 – 308. [Перевод: *Хакен А.* Труднорешаемость резолюций // Кибернетический сборник. Новая серия. М.: Мир, 1991. Вып. 28. С. 179 – 194.]
6. *Tseitin G.* On the complexity of derivation in propositional calculus // In Studies in Constructive Mathematics and Mathematical Logic, part 2. 1968. P. 115 – 125. Reprinted in J. Siekmann and G. Wrightson (editors). Automation of reasoning. Berlin: Springer Verlag, 1983. V. 2. P. 466 – 483.
7. *Семенов А.А.*, *Буранов Е.В.* Погружение задачи криптоанализа симметричных шифров в пропозициональную логику // Вычислительные технологии (совместный выпуск с журналом «Региональный вестник Востока»). 2003. Т. 8. С. 118 – 126.

8. *Семенов А.А.* О сложности обращения дискретных функций из одного класса // Дискретный анализ и исследование операций. 2004. Т. 11. № 4. С. 44 – 55.
9. *Simon J.* On Some Central Problems in Computational Complexity: Doctoral Thesis, Dept. of Computer Science, Cornel University, Ithaca, NY, 1975.
10. *Гэри М.*, *Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
11. *Семенов А.А.* Логико-эвристический подход в криптоанализе генераторов двоичных последовательностей // Труды Междунар. науч. конф. ПАВТ'07. Челябинск: ЮУрГУ, 2007. Т. 1. С. 170 – 180.
12. *Cook S.A.* The complexity of theorem-proving procedures // Proc. 3rd Ann. ACM Symp. on Theory of Computing, Association for Computing Machinery, Ohio, 1971. P. 151 – 159. [Перевод: *Кук С.А.* Сложность процедур вывода теорем // Кибернетический сборник. Новая серия. М.: Мир, 1975. Вып. 12. С. 5 – 15.]
13. *Буранов Е.В.* Программная трансляция процедур логического криптоанализа симметричных шифров // Вестник ТГУ. Приложение. 2004. № 9 (1). С. 60 – 65.
14. *Буров П.С.*, *Игнатьев А.С.*, *Отпущенников И.В.* Программная трансляция алгоритмов в логические выражения в задачах диагностирования дискретных систем // Материалы IX школы-семинара «Математическое моделирование и информационные технологии». Иркутск, 2007. С. 33 – 35.
15. *Семенов А.А.*, *Заикин О.С.* Неполные алгоритмы в крупноблочном параллелизме комбинаторных задач // Труды Междунар. науч. конф. ПАВТ'08. СПб., 2008. С. 232 – 244.
16. *Заикин О.С.* Пакет прикладных программ Distributed-SAT. Свидетельство об официальной регистрации программы ЭВМ № 2008610423, опубл. 23.01.2008.
17. *Феоктистов А.Г.*, *Сидоров И.А.* Языковые средства описания распределенных вычислений в инструментальном комплексе DISCOMP // Труды Междунар. науч. конф. ПАВТ'08. СПб., 2008. С. 488 – 493.
18. MiniSat [http://minisat.se/MiniSat.html]
19. *Biryukov A.*, *Shamir A.*, *Wagner D.* Real Time Cryptanalysis of A5/1 on a PC. Fast Software Encryption Workshop 2000. April 10 – 12, 2000. New-York City.
20. *Суперкомпьютерный* центр ИДСТУ СО РАН [http://www.mvs.icc.ru]
21. *Научно-исследовательский* вычислительный центр МГУ им. М.В. Ломоносова [http://parallel.ru/cluster/]
22. *Список* суперкомпьютеров топ-50 [http://supercomputers.ru/]
23. *Межрегиональный* вычислительный центр ТГУ [http://skif.tsu.ru]