Однако с ростом n — длины блока — увеличивается и размер памяти, который необходим для хранения таблиц. Это приводит к двум негативным с точки зрения реализации последствиям. Во-первых, если криптосхема реализуется на вычислительных средствах с ограничениями по памяти (например, смарт-карты), то на длину блока накладываются существенные ограничения. Во-вторых, таблицы слишком больших размеров могут целиком не помещаться в кэш-память процессора, что существенно снижает скорость шифрования.

Пусть длина блока $n = m \cdot n' = k \cdot m' \cdot n'$, где n' — длина подстановки. В работе [3] предложено использовать линейные преобразования специального вида. В качестве линейного преобразования L рассматривается композиция $L(x) = A(P(x))$, где P представляет собой перестановку подвекторов длины n', $A = \mathrm{diag}(A_1, A_2, \ldots, A_k)$, $A_i = \overline{A}$, $i = 1, \ldots, k$, $\overline{A} \in GL(m' \cdot n', 2)$. При хранении в виде заранее вычисленных таблиц такое линейное преобразование занимает в k^2 раз меньше памяти, чем произвольное линейное преобразование.

Обычно на практике, за редким исключением, $n' = 8$, поэтому открытым остается вопрос, при каких значениях m' заранее вычисленные таблицы линейного преобразования целиком помещаются в кэш-память. Формально, для хранения таблиц необходимо $2^{n'}(m')^2 \cdot n'$ бит. Так как на ЭВМ помимо программы шифрования может быть запущено еще достаточно большое количество других приложений, то выполнение неравенства $2^{n'}(m')^2 \cdot n' < N_{\mathrm{cache}}$ (где N_{cache} — размер кэш-памяти процессора) является необходимым, но не достаточным условием полного размещения таблицы в кэш-памяти.

С целью определения наилучших с точки зрения скорости шифрования и количества тактов процессора, необходимых для обработки одного байта информации, значений $m' \in \{2, 4, 8, 16\}$ при различных значениях $k \in \{2, 3, \ldots, 8\}$ и $n' = 8$ и сопоставимом по криптографическим характеристикам числе итераций были проведены эксперименты, которые показали, что оптимальными значениями являются $m' = k = 2$, $m' = k = 4$ и $m' = k = 8$, которые соответствуют длине блока, равной 32, 128 и 512 бит соответственно. При данных значениях m' и k скорость шифрования в 2,5–1,3 раза выше, чем при любой другой фиксации.

ЛИТЕРАТУРА

1. *Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В.* Основы криптографии. М.: Гелиос АРВ, 2001. 201 с.
2. *Daemen J., Rijmen V.* AES Proposal: Rijndael. http://csrc.nist.gov — Document version 2, 1999.
3. *Rijmen V.* Cryptanalysis and Design of Iterated Block Ciphers. PhD, 2000.

УДК 519.7

О ТРУДОЁМКОСТИ НАПРАВЛЕННОГО ПЕРЕБОРА НЕРАВНОВЕРОЯТНЫХ ВАРИАНТОВ

И. В. Панкратов, О. А. Теплоухова

Решение криптографических задач часто состоит в определении значения ключевого параметра. Мощность множества K значений этого параметра может быть огромной. Кроме того, часто значение самого параметра может быть получено только методом опробования. Этот метод подразумевает последовательный перебор элементов в K до тех пор, пока не будет получено истинное значение. Бывает, что значения в K неравновероятны, и тогда можно осуществить их направленный перебор, начав

с самых вероятных. Трудоёмкость такого перебора определяется как математическое ожидание длины перебора — количества опробованных значений, вычисляемое по формуле $M = \sum_{j=1}^{H} q_j \cdot j$, где H — мощность множества K, q_j — вероятность j-го значения и $q_1 \geqslant q_2 \geqslant \ldots \geqslant q_H$.

Как правило, вычислить трудоёмкость перебора по этой формуле невозможно в силу необъятности множества значений. Поэтому необходимо строить легковычислимую оценку трудоёмкости. В данной предлагается использовать следующий вероятностный алгоритм.

Разобъём множество K на два подмножества одинаковой мощности $h = H/2$. Обозначим M_0 трудоёмкость перебора значений в K, M_1 — трудоёмкость перебора значений в блоке разбиения (будем предполагать, что она одинакова для обоих блоков), p_0 — вероятность того, что искомое значение находится в первом блоке, $p_1 = 1 - p_0$ — вероятность того, что искомое значение находится во втором блоке. Пусть $p_0 \geqslant p_1$.

Любая сумма вида $\sum_{j=1}^{H} q_j \cdot i_j$, где $(i_1, \ldots, i_H)$ есть перестановка чисел $1, \ldots, H$, является оценкой сверху для величины M.

Если предположить, что значения в блоках перебираются последовательно, то есть сначала все элементы первого (более вероятного) блока, затем — все элементы второго, то получим следующую оценку:

$$M_0 = M_1 + p_1 h. \tag{1}$$

Если по очереди выбирать по одному элементу из каждого блока, то оценка будет следующей:

$$M_0 = 2M_1 - p_0. \tag{2}$$

Первая оценка будет близка к истинной в случае, когда вероятность p_1 близка к нулю; вторая — если вероятности p_0 и p_1 примерно равны. Обе оценки не позволяют объективно оценить трудоёмкость в случае, когда обе вероятности значительные, но в разы различаются между собой.

Рассмотрим следующий способ перебора значений: в первую очередь опробуем n элементов первого блока, затем будем по очереди выбирать оставшиеся элементы первого блока и $(h - n)$ элементов второго, затем — остаток второго блока. Тогда оценка имеет вид

$$M_0(n) = \begin{cases} M_1 + T + t_1, & \text{если } t_1 > t_2; \\ M_1 + T + t_3 & \text{иначе,} \end{cases} \tag{3}$$

где

$$T = p_0 \frac{(M_1 - 1)(h^2 + h - 2hn + n^2 - n - 2)}{h^2 + h + 2n - 2};$$

$$t_1 = \sqrt{k(k+1)} - \frac{1}{2} - M_1;$$

$$t_2 = \frac{k-1}{2}\sqrt{\frac{k}{k+1}};$$

$$t_3 = \frac{1}{8}(k-1)\left(2M_1 + 1 - \sqrt{(2M_1+1)^2 - 8k}\right);$$

$$k = h - n.$$

В формуле (3) можно варьировать параметр n и выбирать такой, при котором значение $M_0(n)$ наименьшее. Этот подход оправдал себя, во многих случаях выдавая в эксперименте более точные результаты по сравнению с формулами (1), (2).

Величина M_1 в формулах (1) – (3) находится по одному из блоков разбиения, причем с вероятностью p_0 — по первому блоку и с вероятностью p_1 — по второму. Вычисляется M_1 аналогично M_0 — разбиением соответствующего блока на два подмножества, и так до тех пор, пока не получатся одноэлементные блоки, трудоёмкость перебора элементов в которых равна 1.

Описанные действия предлагается повторить многократно и в качестве результата взять среднее арифметическое полученных оценок. Практические исследования показывают, что погрешность метода составляет до 17 %.

УДК 519.178

МИНИМИЗАЦИЯ ФУНКЦИОНАЛОВ, АССОЦИИРОВАННЫХ С ПРОБЛЕМАМИ ГАМИЛЬТОНОВ ЦИКЛ И ИЗОМОРФИЗМ ГРАФОВ

Р. Т. Файзуллин

В работе предложена конструкция, сводящая задачи ГАМИЛЬТОНОВ ЦИКЛ и ИЗОМОРФИЗМ ГРАФОВ к проблеме поиска глобального экстремума для двух семейств функционалов специального вида, и предложены оригинальные алгоритмы минимизации.

Пронумеруем вершины графа простыми числами, образующими сверхвозрастающую последовательность $R = (r_1, r_2, \ldots)$, где $r_1 = 3$ и $r_{p+1} = 2r_p + s_p, p = 1, 2, \ldots$ Пусть Υ_i — множество вершин графа, соседних с вершиной номер i, где i — одно из чисел r_p.

В этом случае верна теорема 1.

Теорема 1. Если в графе существуют s гамильтоновых циклов с номерами вершин в них $t_1^r, \ldots, t_n^r$, $1 \leqslant r \leqslant s$, то тогда для любого $m \geqslant 2$ верно следующее.

1. Глобальный минимум функционалов

$$S_m(x_1, \ldots, x_n) = \sum_{i=1}^{n} \prod_{j=1}^{n} \prod_{l,p \in \Upsilon_i} ((i + l + p - x_{j-1} - x_j - x_{j+1})^2 + \\ + (mi + l + p - x_{j-1} - mx_j - x_{j+1})^2);$$

$$D_m(x_1, \ldots, x_n) = \sum_{i=1}^{n} \prod_{j=1}^{n} \prod_{l,p \in \Upsilon_i} ((i/lp - x_j/x_{j-1}x_{j+1})^2 + (i^m/lp - x_j^m/x_{j+1}x_{j-1})^2),$$

равный нулю, достигается тогда и только тогда, когда граф гамильтонов.

2. Глобальный минимум достигается только на тех векторах, компоненты которых, рассматриваемые как натуральные числа, являются номерами какого-либо гамильтонова цикла в порядке его прохождения: $x_1 = t_1^r, \ldots, x_n = t_n^r$.

Предложен итерационный алгоритм нахождения стационарных точек функционалов S_m с использованием метода последовательных приближений с инерцией и проведены численные эксперименты на кластере Омского государственного технического университета, показавшие перспективность данного подхода в наиболее интересном случае, когда s мало.

Показано, что выбор в качестве R слабо возрастающих последовательностей (например, возрастающих по логарифмическому закону) и одновременная минимизация нескольких функционалов, с усреднением приближений, позволяет эффективно решать задачу для графов с числом вершин в несколько десятков.