14. *Лидл Р., Нидеррайтер Г.* Конечные поля: В 2 т. / Пер. с англ. – М.: Мир, 1988. – 818 с.

Поступила 20.03.2002

УДК 519.6:519.712.3:510.52:511.12

Богданов А.М., Зинченко Я.В.

УМНОЖЕНИЕ СВЕРХБОЛЬШИХ ЧИСЕЛ И БЫСТРОЕ ПРЕОБРАЗОВАНИЕ ХААРА

В настоящее время в некоторых алгоритмах, которые применяются в асимметричных криптографических системах, при зашифровании, расшифровании и выполнении вспомогательной задачи по формированию ключей используется операция возведения в степень по модулю простого числа или произведения простых чисел, т.е. вычисление выражений вида:

$$a = b^z \pmod{k}. \tag{1}$$

Например, в системе RSA зашифрование сводится к вычислению:

$$c = m^e \pmod{n}, \tag{2}$$

где c – зашифрованное сообщение, m – открытое сообщение, e – ключ зашифрования и $n = p \cdot q$ – модуль. Расшифрование, соответственно, сводится к вычислению:

$$m = c^d \pmod{n}, \tag{3}$$

где d – ключ расшифрования.

В системе экспоненциального ключевого обмена открытый ключ формируется по правилу:

$$Y = \alpha^x \pmod{q}, \tag{4}$$

где Y – открытый ключ, α – фиксированный примитивный элемент поля $GF(q)$, x – ключ зашифрования и q – модуль.

При выполнении операции модулярного возведения в степень на универсальных ЭВМ требуется большое количество вычислений (в соотношениях (1)-(4) параметры представляют собой целые многоразрядные числа размерностью 512-4096 бит), поэтому асимметричные криптосистемы, в которых используется операция модулярного возведения в степень, несмотря на все свои достоинства, не являются эффективными с точки зрения скоростных характеристик при зашифровании и расшифровании сообщений большой длины. Так, например, быстродействие программной реализации RSA примерно в 100 раз ниже, чем быстродействие программной реализации DES. Кроме того, последние достижения в теории криптоанализа асимметричных алгоритмов (дискретное логарифмирование в конечных полях и факторизация больших чисел) заставляют разработчиков увеличивать размерности параметров систем, что еще больше снижает скорость их работы [1].

Учитывая вышесказанное, существует необходимость в выделении ряда методов и приемов, которые позволили бы минимизировать количество вычислений при выполнении модулярного возведения в степень и тем самым увеличить скорость работы асимметричных криптосистем.

Рассмотрим пример вычисления модулярной экспоненты Y в соотношении (4).

Пусть $x = 9$, тогда:

$$Y = \alpha^9 (\mathrm{mod}\ q) = \alpha \cdot \alpha \cdot \alpha \cdot \alpha \cdot \alpha \cdot \alpha \cdot \alpha \cdot \alpha \cdot \alpha (\mathrm{mod}\ q).$$

С вычислительной точки зрения приведенная схема нахождения Y не является оптимальной, поэтому на практике чаще всего используются две следующие схемы:

$$Y = \alpha^9 (\mathrm{mod}\ q) = \left(\left(\left(\alpha^2 \right)^2 \right)^2 \cdot \alpha \right) (\mathrm{mod}\ q),$$

а также $$Y = \alpha^9 (\mathrm{mod}\ q) = \left(\left(\left(\alpha^2 (\mathrm{mod}\ q) \right)^2 (\mathrm{mod}\ q) \right)^2 (\mathrm{mod}\ q) \cdot \alpha \right) (\mathrm{mod}\ q).$$

Таким образом, принимая во внимание тот факт, что модулярное возведение в степень представляет собой последовательность возведений в квадрат (умножений) и модулярных редукций (модулярную редукцию), можно выделить три основных направления решения проблемы минимизации количества вычислений при его выполнении:

- минимизация количества умножений;
- минимизация числа шагов (битовых операций) при выполнении каждого отдельного умножения;
- минимизация количества вычислений при выполнении модулярных редукций.

Остановимся более подробно на минимизации числа шагов при выполнении каждого отдельного умножения.

Известно, что число шагов (битовых операций), необходимых для умножения двух m-разрядных чисел "в столбик", равняется m^2. Однако, нашли широкое применение методы, позволяющие вычислить требуемое произведение быстрее, чем за m^2 шагов. Это метод Карацубы со сложностью $O\left(m^{1.59}\right)$, модулярный метод, имеющий сложность $O\left(m^{1.63}\right)$, и другие. Алгоритм Шенхаге-Штрассена [2] является асимптотически самым быстрым из известных и позволяет умножить два m-разрядных числа за $m \log m \log\log m$ шагов. В основу этого алгоритма положена идея использования теоремы о дискретной свертке двух функций, так как произведение многоразрядных чисел без учета переносов является дискретной циклической сверткой двух сомножителей.

Рассмотрим непосредственно сам алгоритм, как он изложен в [3].

Пусть $T = 2^\gamma$, $\gamma > 0$ – целое число, и необходимо вычислить произведение двух T-разрядных чисел u и v; $R = u \cdot v$. В результате умножения получается $2T$-разрядное двоичное число. Выберем натуральные числа l и p такие, что $l \cdot 2^p \geq 2T$.

Разобьем числа u и v на $K = 2^p$ блоков длины l:

$$u = \sum_{j=0}^{2^p-1} u_j \cdot 2^{jl}, \qquad v = \sum_{j=0}^{2^p-1} v_j \cdot 2^{jl}, \quad (5)$$

где $0 \leq u_j,\ v_j \leq 2^l - 1$ и $u_j = v_j = 0$ для $j \geq 2^{p-1}$. (6)

Тогда их произведение R имеет вид:

$$R = u \cdot v = \left(\sum_{j=0}^{2^p-1} u_j \cdot 2^{jl} \right) \cdot \left(\sum_{j=0}^{2^p-1} v_j \cdot 2^{jl} \right) = \sum_{\tau=0}^{2^p-1} r_\tau \cdot 2^{\tau l}, \quad (7)$$

где r_τ – циклическая свертка u_ρ и v_δ (при выполнении условия (6)):

$$r_\tau = \sum_{\rho+\delta=\tau(\mathrm{mod}\,2^p)} u_\rho \cdot v_\delta \,. \qquad (8)$$

Из соотношений (5)-(8) видно, что суть всего алгоритма действительно сводится к вычислению дискретной циклической свертки. Поскольку такая свертка дает основной вклад в оценку сложности алгоритма, то для эффективного ее вычисления используется алгоритм быстрого преобразования Фурье (БПФ).

Пошаговое описание алгоритма умножения $R = u \cdot v$ имеет следующий вид:

входные данные: u, v.

выходные данные: R.

1. Вычислить с помощью алгоритма БПФ дискретные преобразования Фурье:

$$\tilde{u}_s = \sum_{j=0}^{2^p-1} u_j \cdot w_p^{js}\,,\ \tilde{v}_s = \sum_{j=0}^{2^p-1} v_j \cdot w_p^{js}\,,\ s = \overline{0,\ K-1}\,,\ \text{где } w_p = \exp\left(2\pi / 2^p\right),\ i = \sqrt{-1}\,.$$

2. Вычислить произведение: $\tilde{c}_s = \tilde{u}_s \cdot \tilde{v}_s$, $s = \overline{0,\ K-1}$.

3. Вычислить с помощью алгоритма БПФ r_τ как обратное дискретное преобразование Фурье вектора $\tilde{c}_s$:

$$r_\tau = \frac{1}{2^p} \sum_{s=0}^{2^p-1} \tilde{c}_s \cdot w_k^{-s\tau}\,,\ \tau = \overline{0,\ K-1}\,.$$

4. Принять $R_0 = r_0^{\text{мл}}$ (где $r_s^{\text{ст}}$, $r_s^{\text{мл}}$ – соответственно старшее и младшее слово s-й компоненты свертки r_τ).

5. Вычислить $r_{s+1} = r_s^{\text{ст}} + r_{s+1}$, $s = \overline{0,\ K-2}$.

6. Принять $R_{s+1} = r_{s+1}^{\text{мл}}$, $s = \overline{0,\ K-2}$.

7. Вычислить $R = r_{K-1}^{\text{ст}}$.

Дальнейшее уменьшение порядка сложности или мультипликативной составляющей вышеописанного алгоритма умножения многоразрядных чисел возможно за счет применения более эффективных алгоритмов вычисления дискретной циклической свертки.

Вообще говоря, среди множества существующих алгоритмов вычисления циклической и линейной сверток можно выделить алгоритмы с непосредственным вычислением свертки во временной области и с использованием обобщенного дискретного преобразования Фурье (ОДПФ). В свою очередь, алгоритмы, использующие ОДПФ, базируются на применении БПФ, цифровых фильтров, теоретико-числовых преобразований (ТЧП), определенных в различных кольцах и алгебраических структурах, полиномиальной алгебры (ректангулярные и полиномиальные преобразования) [4].

В работе [5] описан и проанализирован эффективный алгоритм умножения многоразрядных чисел, в основу которого положен разработанный в [6] алгоритм вычисления циклической свертки r_τ, основанный на трансформации пространств исходных данных $N = 2^n$ в пространства соответствующих коэффициентов преобразования Уолша, размерностью $2 \cdot 3^{n-1}$, и их комбинаций в виде сумм и разностей. Для эффективного вычисления коэффициентов разложения используется алгоритм быстрого преобразования Уолша (БПУ).

Рассмотрим алгоритм умножения многоразрядных чисел, основанный на быстром специальном алгоритме вычисления дискретной циклической свертки r_τ, идея построения которого заключается в приведении, с использованием преобразования Хаара, циркулянтной матрицы к блочно-диагональному виду и сведении, таким образом, определения

циклической и линейной сверток к вычислению ряда знаковых сверток коэффициентов разложения пространств исходных данных $N = 2^n$ по системе Хаара [7]. Для эффективного вычисления коэффициентов разложения используется алгоритм быстрого преобразования Хаара (БПХ).

В соответствии с работой [7], введем некоторые обозначения.

Обозначим циклическую свертку $r_{xy}(x \circ y)$ двух векторов x и y длины $N = 2^n$:

$$r_{xy}(k) = r_\tau = \sum_{m=0}^{N-1} x_m y_{k \underset{N}{\oplus} m}, \quad k = \overline{0,\ N-1}, \tag{9}$$

где $\underset{N}{\oplus}$ – сложение по модулю N, а также их линейную свертку $S_{xy}(x \Diamond y)$:

$$S_{xy}(k) = \sum_{m=0}^{N-k-1} x_m y_{m+k}. \tag{10}$$

Так как свертки (9) и (10) легко выражаются друг через друга с помощью соотношений:

$$r_{xy}(m) = S_{xy}(m) + S_{xy}(N-m-1), \tag{11}$$

$$S_{xy}(m) = r_{\tilde{x}\tilde{y}}(m), \qquad m = \overline{0,\ N-1}, \tag{12}$$

где $\tilde{x}_r = \begin{cases} x_r, & r < N \\ 0, & r \geq N \end{cases}$, $\tilde{y}_r = \begin{cases} y_r, & r < N \\ 0, & r \geq N \end{cases}$ $r = \overline{0,\ 2N-1}$, то алгоритмы, полученные для циклической свертки, легко преобразуются в алгоритмы линейной свертки и обратно.

Определим знаковую свертку $P_{xy}(x \nabla y)$ векторов x и y длины $N = 2^n$ следующим образом:

$$P_{xy}(k) = \sum_{m=0}^{N-1} x_k y_{k \underset{N}{\oplus} m} Sign\left[\left(m \underset{N}{\oplus} k\right) - m\right], \tag{13}$$

где $Sign(x) = \begin{cases} 1, & x \geq 0, \\ -1, & x < 0. \end{cases}$

Запишем циклическую свертку (9) в матричной форме:

$$r_{xy} = r_\tau = x_c y, \tag{14}$$

где x_c – циркулянтная матрица; представим ее в виде:

$$x_c = \sum_{m=0}^{N-1} x_m D^m, \tag{15}$$

где D – матрица циклического сдвига на одну позицию.

Характеристический полином $d(\lambda)$ матрицы D, равный $(\lambda^N - 1)$, при $N = 2^N$ разлагается над $\Re$ на множители:

$$d(\lambda) = (\lambda - 1)(\lambda + 1)(\lambda^2 + 1), \ldots, (\lambda^{N/2} + 1), \tag{16}$$

вследствие чего справедливо утверждение о том, что матрица D приводима над $\Re$.

Для понимания сути последующих рассуждений, приведем доказанную в [7] теорему о преобразовании циркулянтной матрицы к блочно-диагональному виду с помощью преобразования Хаара.

Теорема. Матрица циклического сдвига на один разряд преобразованием Хаара приводится к блочно-диагональному виду, т.е.:

$$HDH^{-1} = A = diag\{S_0 - S_0 S_1 S_2 S_4 \ldots S_{N/2}\}, \tag{17}$$

где S_i – матрица, размерностью $2^i \times 2^i$, с элементами

$$[S_i]_{lk} = \begin{cases} 1, & l = k+1, \\ -1, & l = 2^i - 1,\ k = 0, \quad S_0 = 1, \\ 0, & \end{cases}$$

H – матрица преобразования Хаара.

Из (17) следует, что:

$$D = H^{-1} A H. \tag{18}$$

Подставим (18) в (15):

$$x_c = \sum_{m=0}^{N-1} x_m \left(H^{-1} A H\right)^m = \sum_{m=0}^{N-1} x_m H^{-1} A^m H. \tag{19}$$

Умножив (19) слева на H и справа на H^{-1}, получим:

$$H x_c H^{-1} = \sum_{m=0}^{N-1} x_m A^m. \tag{20}$$

Таким образом, из (20) следует, что преобразование Хаара приводит циркулянтную матрицу к блочно-диагональному виду, что и требовалось доказать.

Соотношение (20) также может быть записано следующим образом:

$$H x_c H^{-1} = h_0 \times \prod_{i=0}^{N/4} \sum_{k=0}^{2^i-1} h_{2^i+k} s_i^k, \tag{21}$$

где h_i – коэффициенты разложения вектора x по функциям Хаара; $\times$ означает прямое произведение матриц.

Перейдем непосредственно к быстрому алгоритму вычисления дискретной циклической свертки. Введем некоторые дополнительные обозначения.

Конечные временные ряды $x_0, \ldots, x_{N-1}$ и $y_0, \ldots, y_{N-1}$ представим векторами-столбцами:

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix}, \qquad y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{N-1} \end{bmatrix}. \tag{22}$$

Обозначим векторы циклического сдвига в виде:

$$x' = \begin{bmatrix} -x_{N-1} \\ x_0 \\ x_1 \\ \vdots \\ x_{N-2} \end{bmatrix}, \qquad x'' = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_0 \end{bmatrix}, \qquad x''' = \begin{bmatrix} x_0 \\ x_{N-1} \\ x_{N-2} \\ \vdots \\ x_1 \end{bmatrix}. \tag{23}$$

Набор матриц E, O, P, S, которые преобразуют вектор длины N в векторы длины $N/2$, определим как:

$$Ex = \begin{bmatrix} x_0 \\ x_2 \\ x_4 \\ \\ \vdots \\ \\ x_{N-2} \end{bmatrix}, \quad Ox = \begin{bmatrix} x_1 \\ x_3 \\ x_5 \\ \\ \vdots \\ \\ x_{N-1} \end{bmatrix}, \quad Px = \begin{bmatrix} x_0 + x_1 \\ x_2 + x_3 \\ x_4 + x_5 \\ \\ \vdots \\ \\ x_{N-2} + x_{N-1} \end{bmatrix}, \quad Sx = \begin{bmatrix} x_0 - x_1 \\ x_2 - x_3 \\ x_4 - x_5 \\ \\ \vdots \\ \\ x_{N-2} - x_{N-1} \end{bmatrix}. \tag{24}$$

Отметим, что:

$$Px = Ex + Ox, \quad Sx = Ex - Ox, \quad Py = Ey + Oy, \quad Sy = Ey - Oy. \tag{25}$$

Поскольку длина вектора после преобразования любым из приведенных выше операторов становится в два раза меньше, то преобразования прекращаются, когда x становится одномерным вектором (числом). Например, если $N = 8$, то $SPOx$ есть скаляр, содержащий 4 элемента:

$$SPOx = SPO\begin{bmatrix} x_0 \\ \\ \vdots \\ \\ x_7 \end{bmatrix} = SP\begin{bmatrix} x_1 \\ x_3 \\ x_5 \\ x_7 \end{bmatrix} = S\begin{bmatrix} x_1 + x_3 \\ x_5 + x_7 \end{bmatrix} = x_1 + x_3 - x_5 - x_7.$$

Справедливы соотношения:

$$EP_{xy} = Ex\nabla Ey + Ox\nabla Oy, \quad OP_{xy} = Ex\nabla Oy + (Ox)'\nabla Ey. \tag{26}$$

Рассмотрим вспомогательные свертки a, b и c вида:

$$\begin{aligned} a &= Ex\nabla Py = Ex\nabla(E+O)y, \\ b &= Sx\nabla Oy = (E-O)x\nabla Oy, \\ c &= \left[(Ox)' - Ex\right]\nabla Ey. \end{aligned} \tag{27}$$

Тогда:

$$EP_{xy} = a - b, \quad OP_{xy} = a + c. \tag{28}$$

Соотношения (27) и (28) сводят вычисление знаковой свертки векторов длиной N к вычислению трех знаковых сверток длиной $N/2$. Так как свертка (13) векторов $x = [x_0]$ и $y = [y_0]$ есть $P_{xy} = [x_0 y_0]$, то для вычисления свертки (13) векторов длиной N потребуется 3^n операций умножения.

Сравнив соотношение (21) с алгоритмом (27)-(28), представляющим собой конечный итерационный процесс, видим, что использование преобразования Хаара дает возможность свести вычисление циклической свертки r_τ к определению ряда знаковых сверток коэффициентов разложения векторов x и y по системе Хаара.

Рассмотрим более эффективный алгоритм вычисления дискретной циклической свертки r_τ, описанный в работе [7].

Представим свертку (9) с использованием соотношений (14) и (21) в операторном виде. Для этого определим матрицы L, V, M, K и G, преобразующие вектор длины N в векторы длины $N/2$:

$$Lx = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N/2-1} \end{bmatrix}, \qquad Vx = \begin{bmatrix} x_{N/2} \\ x_{N/2+1} \\ x_{N/2+2} \\ \vdots \\ x_{N-1} \end{bmatrix},$$

$$Mx = \begin{bmatrix} x_0 + x_{N/2} \\ x_1 + x_{N/2+1} \\ x_2 + x_{N/2+2} \\ \vdots \\ x_{N/2-1} + x_{N-1} \end{bmatrix}, \quad Kx = \begin{bmatrix} x_0 - x_{N/2} \\ x_1 - x_{N/2+1} \\ x_2 - x_{N/2+2} \\ \vdots \\ x_{N/2-1} - x_{N-1} \end{bmatrix}, \quad Gx = \begin{bmatrix} x_0 \cdot x_{N/2} \\ x_1 \cdot x_{N/2+1} \\ x_2 \cdot x_{N/2+2} \\ \vdots \\ x_{N/2-1} \cdot x_{N-1} \end{bmatrix}. \tag{29}$$

Отметим, что:

$$\begin{aligned} &Mx = Lx + Vx, \quad Kx = Lx - Vx, \quad Gx = Lx \cdot Vx, \\ &My = Ly + Vy, \quad Ky = Ly - Vy, \quad Gy = Ly \cdot Vy. \end{aligned} \tag{30}$$

Справедливы соотношения:

$$Lr_{xy} + Vr_{xy} = (L+V)x \circ (L+V)y, \quad Lr_{xy} - Vr_{xy} = (L-V)x \nabla (L-V)y. \tag{31}$$

Из соотношений (31) следует, что для вычисления циклической свертки $r_{xy} = r_\tau$ векторов длиной N необходимо определение циклической и знаковой свертки векторов длиной $N/2$.

Рассмотрим вспомогательные свертки a и b вида:

$$a = Mx \circ My = (L+V)x \circ (L+V)y, \quad b = Gx \nabla Ky = (LV)x \nabla (L-V)y. \tag{32}$$

Тогда:

$$2Lr_{xy} = a + b, \quad 2Vr_{xy} = a - b. \tag{33}$$

Таким образом, требуемая свертка (без множителя 2) получается с помощью линейной комбинации двух вспомогательных сверток: a и b.

При вычислении циклической свертки r_τ по алгоритмам (32)-(33) и (27)-(28), требуется выполнение $(3^n + 1)/2 = (3^{\log_2 N} + 1)/2$ операций умножения.

В том случае когда $x = y$, вышеописанный алгоритм является избыточным. В работе [7] предлагается алгоритм, позволяющий устранить эту избыточность.

Справедливы соотношения:

$$Er_{xx} = Ex \circ Ex + Ox \circ Ox, \quad Or_{xx} = Ex \circ Ox + (Ox \circ Ex)''. \tag{34}$$

Рассмотрим вспомогательные свертки a и b вида:

$$a = Px \circ Px = (E+O)x \circ (E+O)x, \quad b = Ox \circ Ex. \tag{35}$$

Тогда:

$$Er_{xx} = a - b - b''', \quad Or_{xx} = b'' + b'''. \tag{36}$$

Для вычисления циклической свертки по представленному алгоритму требуется выполнение $\left(3^n + 3 + 2^n\right)/4 = \left(3^{\log_2 N} + 3 + 2^{\log_2 N}\right)/4$ операций умножения.

Как отмечалось выше, для обеспечения практической стойкости асимметричных криптосистем длины перемножаемых многоразрядных чисел должны лежать в пределах 512-4096 бит. Учитывая рост производительности современных компьютеров, длины сомножителей могут быть увеличены до 8192 бит. Так как сомножители разбиваются на блоки (по 4, 8 или 16 бит), и к каждому из блоков добавляется слева такое же количество нулей (таким способом осуществляется переход от циклической свертки к линейной), то число разрядов, необходимых для представления каждого из сомножителей, увеличивается вдвое. Например, при перемножении двух чисел длиной 8192 бита, реально перемножаются последовательности в 16384 бита. Последовательность такой длины может быть представлена 512-ю блоками по $32 = 16 + 16$ разряда ($512 = 2^9$), 1024-я блоками по $16 = 8 + 8$ разрядов ($1024 = 2^{10}$) или 2048-ю блоками по $8 = 4 + 4$ разрядов ($2048 = 2^{11}$).

Таким образом, практический интерес для криптографических приложений представляют значения n от 5 до 11. Поэтому произведем анализ эффективности алгоритма вычисления свертки в диапазоне значений n от 5 до 13.

В сводной табл. 1 приведено количество операций умножения Q^*, необходимых для вычисления дискретной циклической свертки r_t различными алгоритмами: классическим (Q^*_{ST}), с использованием БПФ (Q^*_{FFT}), с использованием БПУ (Q^*_{FFW}) и с использованием БПХ (Q^*_{FFH1} и Q^*_{FFH2}).

Таблица 1

Сложность алгоритмов вычисления циклической свертки

n	Q^*_{ST} N^2	Q^*_{FFT} $4(n-2)(N-2)+2N$	Q^*_{FFW} $2\cdot 3^{n-1}$	Q^*_{FFH1} $(3^n+1)/2$	Q^*_{FFH2} $(3^n+3+2^n)/4$
13	67108864	376744	1062882	797162	400630
12	16777216	171952	354294	265721	133885
11	4194304	77752	118098	88574	44800
10	1048576	34752	39366	29525	15019
9	262144	15304	13122	9842	5050
8	65536	6608	4374	3281	1705
7	16384	2776	1458	1094	580
6	4096	1120	486	365	199
5	1024	424	162	122	70

Из таблицы видно, что алгоритм вычисления циклической свертки с использованием преобразования Хаара (Q^*_{FFH1}) при $n \le 10$ требует меньше вычислительных затрат, чем алгоритмы с использованием преобразования Фурье и преобразования Уолша. Алгоритм вычисления циклической свертки с использованием преобразования Хаара (Q^*_{FFH2}) требует меньше вычислительных затрат, чем алгоритмы с использованием преобразования Фурье и преобразования Уолша, при $n \le 12$.

В заключение вернемся к алгоритму умножения многоразрядных чисел и приведем его пошаговое описание.

Шаг 1. Вычислить дискретную циклическую свертку r_τ с использованием алгоритма (35)-(36).

Шаг 2. Вычислить произведение R согласно соотношению (7), выполняя сдвиги и сложения l-разрядных чисел.

Следует отметить, что на 2-м шаге алгоритма не требуется выполнение операций умножения, а выполняются только сложения и сдвиги.

Используя полученную в [7] оценку сложности алгоритма вычисления дискретной циклической свертки r_τ, можно получить оценку сложности рассматриваемого алгоритма умножения многоразрядных чисел:

$$W^{\times} = Q^{*}_{FFH\,2} = \left(3^{\log_2 N} + 3 + 2^{\log_2 N}\right)/4 = \left(3^{n} + 3 + 2^{n}\right)/4, \tag{37}$$

где $Q^{*}_{FFH\,2}$ – сложность алгоритма (35)-(36) вычисления свертки r_τ.

Сравним разработанный алгоритм умножения больших чисел с алгоритмом, описанным в [3], в котором для вычисления дискретной циклической свертки r_τ используется модифицированный алгоритм БПФ с предварительной заготовкой элементов матрицы преобразования. Сложность алгоритма [3] равна:

$$T^{\times} = 3K\left(\log_2 K - 1\right) - 16 = 3N\left(\log_2 N - 1\right) - 16 = 3 \cdot 2^{n}\left(n - 1\right) - 16, \tag{38}$$

где $K = 2^{p} \sim N = 2^{n}$ – количество блоков, на которые разбиваются числа-сомножители (длина временного ряда).

Результаты сравнения алгоритмов умножения приведены в сводной табл. 2, содержащей количество умножений, необходимых для вычисления произведения многоразрядных чисел каждым из них.

Таблица 2

Сложность алгоритмов умножения многоразрядных чисел

n	3	4	5	6	7	8	9	10	11	12	13
$T^{\times}$	32	128	368	944	2288	5360	12272	27632	61424	135152	294896
$W^{\times}$	10	25	70	199	580	1705	5050	15019	44800	133885	400630
$T^{\times}$-$W^{\times}$	22	103	298	745	1708	3655	7222	12613	16624	1267	-105734

Анализ таблицы показывает, что эффективность разработанного алгоритма при $n \le 12$ выше эффективности алгоритма, предложенного в [3]. Для указанных значений n при вычислении произведения многоразрядных чисел разработанному алгоритму требуется меньшее количество умножений, чем алгоритму, использующему БПФ.

Эффективность разработанного алгоритма определяется коэффициентом эффективности h по соотношению:

$$h = \frac{T^{\times} - W^{\times}}{T^{\times}} \cdot 100\% . \qquad (39)$$

На рис. 1 изображен график зависимости коэффициента эффективности разработанного алгоритма, выраженного в процентах, от значений n. Из графика видно, что максимальная экономия по числу умножений достигается при малых n ($h = 82\%$ при $n = 5$), поэтому данный алгоритм наиболее целесообразно использовать при умножении 512 битных чисел.

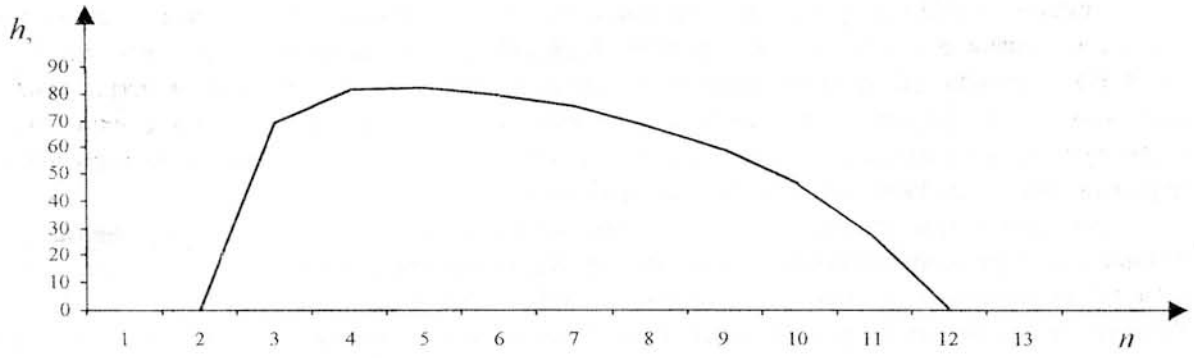

Рис.1. Эффективность разработанного алгоритма

Список литературы

1. *Чмора А.Л.* Современная прикладная криптография. – М.: Гелиос, 2001. – 256 с.
2. *Кнут Д.* Искусство программирования для ЭВМ. Т.2. – М.: Мир, 2001. – 730 с.
3. *Задирака В.К., Мельникова С.С.* Быстрое умножение многоразрядных чисел с использованием БПФ // Кибернетика и системный анализ. – 1996. – № 3. – С. 63-67.
4. *Блейхут Р.* Быстрые алгоритмы цифровой обработки сигналов. – М.: Мир, 1989. – 448 с.
5. *Задирака В.К., Мельникова С.С.* Анализ сложности алгоритма умножения сверхбольших чисел на основе коэффициентов Уолша // Кибернетика и системный анализ. – 2001. – № 6. – С. 99-110.
6. *Pitassi D. A.* Fast convolution using the Walsh transform // Appl. of Walsh Funktions. – 1971. – P. 130-133.
7. *Садыхов Р., Шаренков А.* Алгоритмы ускоренной свертки // Автоматика. – 1986. – №3. – С. 71-75.

Поступила 27.08.2002