УДК: 519.6

А.Н. Терещенко

ВОЗВЕДЕНИЕ В СТЕПЕНЬ n И ВЫЧИСЛЕНИЕ КОРНЯ СТЕПЕНИ n БОЛЬШИХ ЧИСЕЛ НА ОСНОВЕ РЕКУРРЕНТНЫХ СООТНОШЕНИЙ

Описываются метод возведения в степень n и метод вычисления корня степени n больших чисел на основе рекуррентных соотношений (n>1). Описанные методы являются обобщением методов возведения в степени 2, 3 и методов вычисления квадратного и кубического корней. Метод вычисления корней степени n больших чисел основан на методе вычисления степени n. Метод вычисления корней степени n не использует операции деления. При возведении большого числа X в степень n или вычислении корня степени n одновременно с нахождением результата вычисляются значения всех степеней до n-1. Методы являются побитовыми, где на каждой итерации обрабатывается следующий бит (n бит − при вычислении корня) исходной битовой последовательности.

Введение

Криптографические операции основаны на элементарных операциях таких, как сложение, умножение, возведение в степень больших чисел, получение остатка по модулю большого числа и т. д. Длина секретного ключа влияет на криптостойкость ассиметричных криптографических программно-аппаратных комплексов. В свою очередь длина ключа влияет на быстродействие всей системы, поэтому выбор длины ключа зависит от быстродействия каждой криптографической операции, а значит, зависит и от быстродействия элементарных арифметических операций, на основе которых построены криптографические операции. Оптимизации быстродействия элементарных операций таких, как умножение больших чисел, возведение в степень по модулю большого числа и т. д. уделяется много внимания [1–4].

Цель работы – обобщение для любой степени n выше третей метода [5] возведения в квадрат и куб большого числа, а также метода вычисления квадратного и кубического корней больших чисел без использования каких-либо операций деления (практический диапазон > 1024 битовых разрядов).

Постановка задачи

Пусть $m, n > 0$ – целые числа, положительное число U представлено m битами, а число W представлено $n \cdot m$ битами:

$$U_m = \sum_{j=0}^{m-1} u_j 2^j \ , \ W_{n \cdot m} = \sum_{j=0}^{n \cdot m-1} w_j 2^j \ .$$

Возведение числа U_m длиной в m бит в степень n и вычисление корня степени n числа $W_{n \cdot m}$ длиной в $n \cdot m$ бит можно представить следующим образом:

$$(U_m)^n = (U_{m-1} \cdot 2 + u_{m-1})^n =$$

$$= \sum_{i=0}^{n} C_n^i \cdot (U_{m-1})^{n-i} \cdot (u_{m-1})^i \cdot 2^i \ ,$$

$$\sqrt[n]{W_{n \cdot m}} = R_m + \sqrt[n]{(V_m)^n} =$$

$$= R_m + \sqrt[n]{(V_{m-1} \cdot 2 + v_{m-1})^n} =$$

$$= R_m + \sqrt[n]{\sum_{i=0}^{n} C_n^i \cdot (V_{m-1})^{n-i} \cdot (v_{m-1})^i \cdot 2^i} \ ,$$

где $C_n^i = \dfrac{n!}{i! \cdot (n-i)!}$, $U_{m-1} = \sum_{j=0}^{m-2} u_j 2^j$,

$$V_{m-1} = \sum_{j=0}^{m-2} v_j 2^j \ , \ R_m = \sum_{j=0}^{m-1} r_j 2^j \ .$$

На основании вышеприведенных методов построим итерационные побитовые алгоритмы.

1. Возведение в любую степень n $(U_m)^n$ числа U_m длиной в m бит.

2. Вычисление целочисленных корня V_m и остатка R_m с длинами в m бит любой степени n числа $W_{n \cdot m}$ длиной в $n \cdot m$ бит без использования деления.

Провести анализ сложности по числу битовых операций этих алгоритмов.

Определения и обозначения

Большими числами будем называть числа, длина которых в битах превышает разрядность машинного слова, то есть свыше 32 бит. Для реализации операций с такими числами обычный набор команд процессора недостаточен и необходимо использование специальных алгоритмов и методов.

В данной работе большие числа представлены в виде битовых последовательностей. Далее в тексте под обработкой битовых последовательностей подразумевается обработка больших чисел.

Для лучшего представления материала предлагается обратная нумерация битов в последовательности. В этом случае индекс самого старшего бита равен нулю, а индекс самого младшего бита равен длине последовательности в битах минус 1. Такое представление удобно, так как анализ начинается со старшего бита, а длина последовательности не важна, так как к проанализированной битовой последовательности могут добавляться младшие биты. Такой подход является одним из преимуществ использования данного метода, так как он дает возможность получить при необходимости точность вычисления больше заданной при нахождении корней больших степеней от больших чисел, используя один и тот же алгоритм.

Таким образом, битовая последовательность записывается в обычном порядке, т. е. самый старший бит располагается слева, и в этом случае младшие биты добавляются с правой стороны. Далее приведен пример битовой последовательности длиной в 8 бит:

$$x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7,$$

что равносильно выражению:

$$x_0 \cdot 2^7 + x_1 \cdot 2^6 + x_2 \cdot 2^5 + x_3 \cdot 2^4 + x_4 \cdot 2^3 + x_5 \cdot 2^2 + x_6 \cdot 2 + x_0.$$

Введем следующие обозначения.

121_{10}, 1111001_2 – представление числа 121 в десятичной и двоичной системах счисления.

i – номер итерации, нумерация начинается с нуля ($i = 0$).

X_i – битовая последовательность.

Y_i – квадрат последовательности X_i.

Z_i – куб последовательности X_i.

$X_i, Y_i, Z_i, i > 0$, – битовые последовательности длиной в $i+1$, $2(i+1)$ и $3(i+1)$ бит, соответственно. Длины последовательностей Y_0 и Z_0 равны 1 и являются исключениями.

X_i, Y_i, Z_i вычисляются на итерации i на основе $X_{i-1}, Y_{i-1}, Z_{i-1}$, полученных на предыдущей $i-1$ итерации.

x_j, y_j, z_j – j-ые биты в последовательностях X, Y, Z соответственно.

$x_i = 0$ – операция сравнения выражений с левой и правой сторон.

Можно записать следующие выражения для квадрата и куба числа X:

$$X = x_0 x_1 x_2 x_3 \ldots,$$

$$X^2 = Y = y_0 y_1 y_2 y_3 y_4 y_5 y_6 y_7 \ldots,$$

$$X^3 = Y \cdot X = Z = z_0 z_1 z_2 z_3 z_4 z_5 z_6 z_7 z_8 z_8 z_{10} z_{11} \ldots$$

$i = \overline{0, m-1}$ – индексу i последовательно присваиваются значения 0, 1, …, $m-1$, m.

$x_i \leftarrow 1$ – операция присвоения значения выражению с левой стороны значения с правой стороны. Необходимо отличать от операции $x_i = 1$, где значения с левой и правой сторон не изменяются по завершению операции. В выражении $i \leftarrow i+1$ значение i увеличивается на 1 после окончания операции. Знак $\leftarrow$ может использоваться несколько раз в одной операции для того, чтобы сократить запись и показать, что одинаковые значения присваиваются различным элементам. Так две операции $X_0 \leftarrow 1$, $Y_0 \leftarrow 1$ можно заменить

одной $X_0 \leftarrow Y_0 \leftarrow 1$. Символы $\leftarrow$ и $=$ также могут использоваться одновременно в одной операции:

$X_2 \leftarrow 2\cdot 2+1 = 5_{10} = 101_2$, что обозначает вычисление выражения $2\cdot 2+1$, результат которого 5_{10}, 101_2 представлен в двух системах счисления, и которое записывается в X_2.

$$\begin{cases} x_1 = 0:\ X_1 \leftarrow X_0 \cdot 2; & Y_1 \leftarrow Y_0 \cdot 4. \\ x_1 = 1:\ X_1 \leftarrow X_0 \cdot 2+1; & Y_1 \leftarrow Y_0 \cdot 4 + 2 \cdot X_1 - 1. \end{cases}$$

– операция вычисления выражений по условию. Если $x_1 = 0$, то выполняются операции $X_1 \leftarrow X_0 \cdot 2$; $Y_1 \leftarrow Y_0 \cdot 4$. Если $x_1 = 1$, то выполняются операции $X_1 \leftarrow X_0 \cdot 2+1$; $Y_1 \leftarrow Y_0 \cdot 4 + 2 \cdot X_1 - 1$.

Порядок выполнения операций в выражениях (формулах, алгоритмах и таблицах) определяется символами ‘;‘ и ‘,‘.

Выражение разбивается на группы символом ‘;‘. Последовательность групп в выражение выполняется в обычном порядке – слева направо. Операции внутри группы разделяются запятой и выполняются в обратном порядке – справа налево. Если операцию обозначить номером ее выполнения, то получим строку, определяющую порядок выполнения операций:

2, 1; 6, 5, 4, 3; 8, 7.

Выражения далее отличаются:

$$i \leftarrow 0;\ x_i \leftarrow 1,\ i \leftarrow 1.$$

$$i \leftarrow 0,\ x_i \leftarrow 1;\ i \leftarrow 1.$$

В первом выражении значение один присвоится элементу с индексом 1, а во втором – с индексом 0, так как порядок выполнения операций отличается:

1; 3, 2.

1, 2; 3.

Возведение в квадрат и вычисление квадратного корня

Возведение в квадрат.

Рассмотрим последовательное получение формул для вычисления битовых последовательностей X_i и Y_i, $i = \overline{0, m-1}$.

Рассмотрим возведения в квадрат 1-битового числа x_0 $(i = 0)$.

Начальная итерация:

$$X_0 = x_0,\ Y_0 = X_0^2 = (x_0)^2 = x_0.$$

Добавим к биту x_0 справа младший бит x_1 и рассмотрим формулы для вычисления квадрата из полученной битовой последовательности $(i = 1)$:

$$X_1 = X_0 x_1 = X_0 \cdot 2 + x_1, \qquad (1)$$

$$\begin{aligned} Y_1 = X_1^2 = (X_0 x_1)^2 = (X_0 \cdot 2 + x_1)^2 = \\ = X_0^2 \cdot 4 + 2 \cdot X_0 \cdot x_1 \cdot 2 + x_1^2 = \\ = Y_0 \cdot 4 + (2 \cdot X_0 \cdot 2 + 1) \cdot x_1 = \\ = Y_0 \cdot 4 + ((2 \cdot X_1 - 2) + 1) \cdot x_1. \end{aligned}$$

Окончательно получаем:

$$Y_1 = Y_0 \cdot 4 + (2 \cdot X_1 - 1) \cdot x_1. \qquad (2)$$

Для различных значений x_1, формулы (1), (2) принимают следующий вид:

$$\begin{cases} x_1 = 0: X_1 \leftarrow X_0 \cdot 2; & Y_1 \leftarrow Y_0 \cdot 4. \\ x_1 = 1: X_1 \leftarrow X_0 \cdot 2+1; & Y_1 \leftarrow Y_0 \cdot 4 + 2 \cdot X_1 - 1. \end{cases} \qquad (3)$$

Обратим внимание, что в формуле (1) коэффициенты 4 и 2 в выражении $Y_1 \leftarrow Y_0 \cdot 4 + 2 \cdot X_1 - 1$ стоят с разных сторон: «4» показывает сдвиг на 2 бита, при этом «2» является коэффициентом при X_1. Такая запись используется далее, чтобы отделить битовые сдвиги от коэффициентов при анализе полученных выражений.

Рассмотрим возведение в квадрат 3-битовой последовательности $x_0 x_1 x_2$, т. е. добавим еще один младший разряд справа $(i = 2)$:

$$\begin{aligned} X_2 = x_0 x_1 x_2 = x_0 x_1 \cdot 2 + x_2 = X_1 \cdot 2 + x_2, \\ Y_2 = X_2^2 = (x_0 x_1 x_2)^2 = (X_1 \cdot 2 + x_2)^2 = \\ = X_1^2 \cdot 4 + 2 \cdot X_1 \cdot x_2 \cdot 2 + x_2^2 = \\ = Y_1 \cdot 4 + (2 \cdot X_1 \cdot 2 + 1) \cdot x_2 = \\ = Y_1 \cdot 4 + (2 \cdot X_2 - 1) \cdot x_2. \end{aligned} \qquad (4)$$

Окончательно получаем:

$$Y_2 = Y_1 \cdot 4 + (2 \cdot X_2 - 1) \cdot x_2.$$

У предыдущего выражения все индексы больше на 1 по сравнению с (2).

По аналогии с формулами (4) для битовой последовательности $x_0x_1x_2x_3$ можно получить формулы для X_3 и Y_3:

$$X_3 = x_0x_1x_2x_3 = X_2 \cdot 2 + x_3,$$

$$Y_3 = Y_2 \cdot 4 + (2 \cdot X_2 - 1) \cdot x_3.$$

Для вычисления Y_{m-1} воспользуемся рекуррентными соотношениями.

Лемма 1. Битовые последовательности Y_i, $i = \overline{0, m-1}$ можно получить на основе X_{i-1}, Y_{i-1} и бита x_i:

$$X_i = x_0x_1...x_{i-1}x_i = X_{i-1} \cdot 2 + x_i,$$

$$Y_i = Y_{i-1} \cdot 4 + (2 \cdot X_i - 1) \cdot x_i,\ i = \overline{0, m-1}.$$

Доказательство. Добавим к битовой последовательности X_{i-1} справа бит x_i по аналогии с (4):

$$X_i = X_{i-1}x_i = X_{i-1} \cdot 2 + x_i,$$

$$Y_i = X_i^2 = (X_{i-1}x_i)^2 = (X_{i-1} \cdot 2 + x_i)^2 =$$

$$= X_{i-1}^2 \cdot 4 + 2 \cdot X_{i-1} \cdot x_i \cdot 2 + x_i^2 =$$

$$= Y_{i-1} \cdot 4 + (2 \cdot X_{i-1} \cdot 2 + 1) \cdot x_i.$$

Окончательно получаем:

$$Y_i = Y_{i-1} \cdot 4 + (2 \cdot X_i - 1) \cdot x_i.$$

Лемма доказана.

Возведение в квадрат битовой последовательности $X_3 = x_0x_1x_2x_3$ можно представить в виде следующих последовательных шагов:

$$y_0y_1 = 0x_0,$$

$$y_0y_1y_2y_3 = y_0y_100 + 0x_001 \cdot x_1,$$

$$y_0y_1y_2y_3y_4y_5 = y_0y_1y_2y_300 + 00x_0x_101 \cdot x_2,$$

$$y_0y_1y_2y_3y_4y_5y_6y_7 =$$

$$= y_0y_1y_2y_3y_4y_500 + 000x_0x_1x_201 \cdot x_3. \quad (5)$$

На каждом шаге второе слагаемое равно нулю, если младший бит нулевой. Более удобно выражения (5) представляются в виде классического столбика, показанного на рис. 1.

$$\begin{array}{cccccccccc}
 & 0 & x_0 & & & & & & & \\
 & & x_0 & 0 & x_1 & & & & & \\
+ & & & x_0 & x_1 & 0 & x_2 & & & \\
 & & & & x_0 & x_1 & x_2 & 0 & x_3 & \\
\hline
 & y_0 & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 &
\end{array}$$

Рис. 1. Схема возведение в квадрат на основе предложенного метода

Согласно выражений (5) строка состоит из нулей, если бит $x_i = 0$. На рис. 2 показан пример возведение в квадрат чисел $11_{10}=1011_2$ и $15_{10}=1111_2$:

$$\begin{array}{cccccccccc}
 & 0 & 1 & & & & & & & x_0 = 1 \\
 & & 0 & 0 & 0 & & & & & x_1 = 0 \\
+ & & & 1 & 0 & 0 & 1 & & & x_2 = 1 \\
 & & & & 1 & 0 & 1 & 0 & 1 & x_3 = 1 \\
\hline
 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 &
\end{array}$$

$$\begin{array}{cccccccccc}
 & 0 & 1 & & & & & & & x_0 = 1 \\
 & & 1 & 0 & 1 & & & & & x_1 = 1 \\
+ & & & 1 & 1 & 0 & 1 & & & x_2 = 1 \\
 & & & & 1 & 1 & 1 & 0 & 1 & x_3 = 1 \\
\hline
 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 &
\end{array}$$

Рис. 2. Примеры вычисления квадрата на основе предложенного метода

Алгоритм 1. Пошаговое описание алгоритма возведения в квадрат (табл. 1).

ВХОД: X – битовая последовательность, m – длина последовательности X ($m \geq 2$).

ВЫХОД: $Y = X^2$ ($Y_i = X_i^2$, $i = \overline{0, m-1}$).

1. Найти самый старший ненулевой бит последовательности $X = x_0x_1x_2x_3...$.

2. Начальная итерация $i \leftarrow 0$; $X_0 \leftarrow Y_0 \leftarrow 1$.

3. $i \leftarrow i + 1$.

4. Найти X_i на основе значения x_i:

$$\begin{cases} x_i = 0: & X_i \leftarrow X_{i-1} \cdot 2. \\ x_i = 1: & X_i \leftarrow X_{i-1} \cdot 2 + x_i. \end{cases}$$

5. Найти Y_i на основе значения x_i (см. (3)):

$$\begin{cases} x_i = 0: & Y_i \leftarrow Y_{i-1} \cdot 4. \\ x_i = 1: & Y_i \leftarrow Y_{i-1} \cdot 4 + 2 \cdot X_i - 1. \end{cases}$$

6. Если бит x_i не последний, то перейти к шагу 3.

Таблица 1. Пример выполнения Алгоритма 1 при вычислении квадрата числа $11_{10} = 1011_2$ (см. рис. 2)

Шаг	Операция
1	Старший бит ненулевой
2	$i \leftarrow 0;\ X_0 \leftarrow Y_0 \leftarrow 1$
3	$i \leftarrow 1$
4;5	Так как $x_1 = 0$, то $X_1 \leftarrow 1 \cdot 2 = 2_{10} = 10_2$; $Y_1 \leftarrow 1 \cdot 4 = 4_{10} = 100_2$
6	Бит x_1 не последний, перейти к шагу 3
3	$i \leftarrow 2$
4; 5	Так как $x_2 = 1$, то $X_2 \leftarrow 2 \cdot 2 + 1 = 5_{10} = 101_2$; $Y_2 \leftarrow 4 \cdot 4 + 2 \cdot 5 - 1 = 25_{10} = 11001_2$
6	Бит x_2 не последний, перейти к шагу 3
3	$i \leftarrow 3$
4;5	Так как $x_3 = 1$, то $X_3 \leftarrow 5 \cdot 2 + 1 = 11_{10} = 1011_2$; $Y_3 \leftarrow 25 \cdot 4 + 2 \cdot 11 - 1 = 121_{10} = 1111001_2$
6	Бит x_3 последний, завершить работу алгоритма

Вычисление квадратного корня. Для вычисления корней небольших чисел с допустимой погрешностью достаточно небольшое число итераций. Для получения корней больших чисел известными методами необходимо многошаговое вычисление с использованием операций умножения, деления больших чисел, вычисления факториала и т. д., что усложняет по времени операцию вычисления корней.

Алгоритм 2 вычисления квадратного корня похож на Алгоритм 1 возведения в квадрат, за исключением того, что происходит последовательное вычитание из последовательности $Y_3 = y_0y_1y_2y_3y_4y_5y_6y_7$ вместо добавления. Схема позволяет найти остаток $R_3 < X_3$, где $X_3 = x_0x_1x_2x_3$, $R_3 = r_0r_1r_2r_3r_4r_5r_6r_7$.

Алгоритм 2. Пошаговое описание алгоритма вычисления квадратного корня.

ВХОД: Y – битовая последовательность, m – длина последовательности Y ($m \geq 3$).

ВЫХОД: $X = \lfloor \sqrt[2]{Y} \rfloor$, $R = Y - X^2$

($X_i = \lfloor \sqrt[2]{Y_i} \rfloor$, $R_i = Y_i - (X_i)^2$, $i = \overline{0, \left\lfloor \frac{m-1}{2} \right\rfloor}$).

R_i – промежуточная битовая последовательность длиной в $2(i+1)$ бит на каждой итерации i.

1. Если длина битовой последовательности Y не делится на 2 нацело, то добавить старший нулевой бит слева. Разбить последовательность

$$Y = y_0y_1 \mid y_2y_3 \mid y_4y_5 \mid y_6y_7 \mid \ldots.$$

на пары бит.

2. Найти самую старшую ненулевую битовую пару.

3. Начальная итерация $i \leftarrow 0;\ X_0 \leftarrow x_0 \leftarrow 1$, $R_0 \leftarrow y_0y_1;\ R_0 \leftarrow R_0 - X_0$ (см. рис. 3, 4).

4. $i \leftarrow i + 1$.

y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	
0	x_0							$\times x_0$
	x_0	0	x_1					$\times x_1$
–		x_0	x_1	0	x_2			$\times x_2$
			x_0	x_1	x_2	0	x_3	$\times x_3$
r_0	r_1	r_2	r_3	r_4	r_5	r_6	r_7	

Рис. 3. Схема вычисления квадратного корня $y_0y_1y_2y_3y_4y_5y_6y_7$

$$
\begin{array}{cccccccc}
0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\hline
0 & 1 & & & & & & \times x_0 = 1 \\
 & 0 & 0 & 0 & & & & \times x_1 = 0 \\
- & & 1 & 0 & 0 & 1 & & \times x_2 = 1 \\
 & & & 1 & 0 & 1 & 0 & 1 \quad \times x_3 = 1 \\
\hline
 & & & & 0 & 0 & 1 & 0
\end{array}
$$

Рис. 4. Пример вычисления квадратного корня числа 123_{10}

5. $X_i \leftarrow X_{i-1} \cdot 2 + 1$.

6. $dY_i \leftarrow 2 \cdot X_i - 1$.

7. $R_i \leftarrow R_{i-1} \cdot 4 + y_{2i} y_{2i+1}$

8. Если $R_i > dY_i$, то $x_i \leftarrow 1$, $R_i \leftarrow R_i - dY_i$, иначе $x_i \leftarrow 0$ (см. рис. 3, 4).

9. Если биты $y_{2i} y_{2i+1}$ не последние, то перейти к шагу 4.

По окончании алгоритма X_i будет содержать целую часть от квадратного корня, а R_i – остаток (табл. 2).

Таблица 2. Пример выполнения Алгоритма 2 при вычислении квадратного корня числа $123_{10} = 1111011_2$ (см. рис. 4)

Шаг	Операция
1	2
1	Добавить старший нулевой бит слева и разбить последовательность $Y = 01 \mid 11 \mid 10 \mid 11$ на пары бит
2	Самая старшая битовая пара ненулевая
3	Начальная итерация $i \leftarrow 0$; $X_0 \leftarrow x_0 \leftarrow 1$; $R_0 \leftarrow 01_2 = 1_{10}$; $R_0 \leftarrow 1 - 1 = 0$
4	$i \leftarrow 1$
5	$X_1 \leftarrow 1 \cdot 2 + 1 = 3_{10} = 11_2$
6	$dY_1 \leftarrow 2 \cdot 3 - 1 = 5_{10} = 101_2$
7	$R_1 \leftarrow 0 \cdot 4 + 3 = 3_{10} = 11_2$
8	Так как $3 < 5$, то $x_1 \leftarrow 0$ ($X_1 = 2$)
9	Биты $y_2 y_4$ не последние, перейти к шагу 4

Окончание табл. 2

1	2
4	$i \leftarrow 2$
5	$X_2 \leftarrow 2 \cdot 2 + 1 = 5_{10} = 101_2$
6	$dY_2 \leftarrow 2 \cdot 5 - 1 = 9_{10} = 1001_2$
7	$R_2 \leftarrow 3 \cdot 4 + 2 = 14_{10} = 1110_2$
8	Так как $14 > 9$, то $x_2 \leftarrow 1$, $R_2 \leftarrow 14 - 9 = 5_{10} = 101_2$
9	Биты $y_4 y_5$ не последние, перейти к шагу 4
4	$i \leftarrow 3$
5	$X_3 \leftarrow 5 \cdot 2 + 1 = 11_{10} = 1011_2$
6	$dY_3 \leftarrow 2 \cdot 11 - 1 = 21_{10} = 10101_2$
7	$R_3 \leftarrow 5 \cdot 4 + 3 = 23_{10} = 10111_2$
8	Так как $23 > 21$, то $x_3 \leftarrow 1$, $R_3 \leftarrow 23 - 21 = 2_{10} = 10_2$
9	Биты $y_6 y_7$ последние, завершить работу алгоритма

Возведение в куб и вычисление кубического корня

Возведение в куб. Рассмотрим последовательное получение формул для вычисления битовых последовательностей X_i и Z_i, $i = \overline{0, m-1}$.

Начнем рассмотрение с возведения в куб 1-битового числа x_0 $(i = 0)$. Начальная итерации будет следующей:

$$X_0 = x_0,\ Y_0 = X_0^2 = (x_0)^2 = x_0,$$
$$Z_0 = X_0^3 = Y_0 \cdot X_0 = (x_0)^3 = x_0.$$

Рассмотрим формулу для получения куба из X_1 $(i = 1)$:

$$Z_1 = X_1^3 = (X_0 x_1)^3 = (X_0 \cdot 2 + x_1)^3 =$$
$$= X_0^3 \cdot 8 + 3 \cdot X_0^2 \cdot x_1 \cdot 4 + 3 \cdot X_0 \cdot x_1^2 \cdot 2 + x_1^3 =$$
$$= Z_0 \cdot 8 + 3 \cdot Y_0 \cdot x_1 \cdot 4 + 3 \cdot X_0 \cdot x_1 \cdot 2 + x_1 =$$
$$= Z_0 \cdot 8 + (3 \cdot Y_0 \cdot 4 + 3 \cdot X_0 \cdot 2 + 1) \cdot x_1 =$$
$$= Z_0 \cdot 8 + \begin{pmatrix} 3 \cdot (Y_0 \cdot 4 + 2 \cdot X_0 \cdot 2 + 1) - \\ -3 \cdot (X_0 \cdot 2 + 1) + 1 \end{pmatrix} \cdot x_1 =$$

$$= Z_0 \cdot 8 + \left(3 \cdot \left(Y_0 \cdot 4 + 2 \cdot X_1 - 1\right) - 3 \cdot X_1 + 1\right) \cdot x_1 .$$

С учетом формулы (2) окончательно получаем:

$$Z_1 = Z_0 \cdot 8 + (3 \cdot Y_1 - 3 \cdot X_1 + 1) \cdot x_1 . \quad (6)$$

Рассмотрим получение куба 3-битовой последовательности $x_0 x_1 x_2$, т. е. добавим еще один младший разряд справа $(i = 2)$:

$$Z_2 = X_2^3 = (x_0 x_1 x_2)^3 = (X_1 \cdot 2 + x_2)^3 =$$

$$= X_1^3 \cdot 8 + 3 \cdot X_1^2 \cdot x_2 \cdot 4 + 3 \cdot X_1 \cdot x_2^2 \cdot 2 + x_2^3 =$$

$$= Z_1 \cdot 8 + \left(3 \cdot Y_1 \cdot 4 + 3 \cdot X_1 \cdot 2 + 1\right) \cdot x_2 =$$

$$= Z_1 \cdot 8 + \begin{pmatrix} 3 \cdot Y_1 \cdot 4 + 3 \cdot X_1 \cdot 4 + 3 - \\ -3 \cdot X_1 \cdot 2 - 3 + 1 \end{pmatrix} \cdot x_2 .$$

Окончательно получаем формулу, похожую на (6):

$$Z_2 = Z_1 \cdot 8 + (3 \cdot Y_2 - 3 \cdot X_2 + 1) \cdot x_2 .$$

По аналогии с предыдущей формулой можно получить формулы для битовой последовательности $x_0 x_1 x_2 x_3$ для Z_3 $(i = 3)$:

$$Z_3 = Z_2 \cdot 8 + (3 \cdot Y_3 - 3 \cdot X_3 + 1) \cdot x_3 .$$

Лемма 2. Битовые последовательности Z_i, $i = \overline{0, m-1}$ можно получить из битовых последовательностей X_{i-1}, Y_{i-1}, Z_{i-1} и бита x_i, используя следующие формулы:

$$X_i = x_0 x_1 ... x_{i-1} x_i = X_{i-1} \cdot 2 + x_i ,$$

$$Y_i = Y_{i-1} \cdot 4 + (2 \cdot X_i - 1) \cdot x_i ,$$

$$Z_i = Z_{i-1} \cdot 8 + (3 \cdot Y_i - 3 \cdot X_i + 1) \cdot x_i , \; i = \overline{0, m-1} .$$

Доказательство. Добавим к битовой последовательности X_{i-1} справа x_i. Выражение $Y_i = Y_{i-1} \cdot 4 + (2 \cdot X_i - 1) \cdot x_i$ следует из леммы 1.

Находим формулу для куба новой последовательности аналогично (6):

$$Z_i = X_i^3 = (X_{i-1} x_i)^3 = (X_{i-1} \cdot 2 + x_i)^3 =$$

$$= X_{i-1}^3 \cdot 8 + 3 \cdot X_{i-1}^2 \cdot x_i \cdot 4 + 3 \cdot X_{i-1} \cdot x_i^2 \cdot 2 + x_i^3 =$$

$$= Z_{i-1} \cdot 8 + \left(3 \cdot Y_{i-1} \cdot 4 + 3 \cdot X_{i-1} \cdot 2 + 1\right) \cdot x_i =$$

$$= Z_{i-1} \cdot 8 + \begin{pmatrix} 3 \cdot Y_i \cdot 4 + 3 \cdot X_{i-1} \cdot 4 + 3 - \\ -3 \cdot X_{i-1} \cdot 2 - 3 + 1 \end{pmatrix} \cdot x_i =$$

$$= Z_{i-1} \cdot 8 + \begin{pmatrix} 3 \cdot \left(Y_i + 2 \cdot X_i \cdot 2 - 1\right) - \\ -3 \cdot \left(X_{i-1} \cdot 2 + 1\right) + 1 \end{pmatrix} \cdot x_i .$$

Окончательно получаем:

$$Z_i = Z_{i-1} \cdot 8 + (3 \cdot Y_i - 3 \cdot X_i + 1) \cdot x_i .$$

Лемма доказана.

Алгоритм 3. Пошаговое описание алгоритма вычисления куба числа.

ВХОД: X – битовая последовательность, m – длина последовательности X $(m \geq 2)$.

ВЫХОД: $Z = X^3$, $Y = X^2$ $(Z_i = X_i^3,\ Y_i = X_i^2,\ i = \overline{0, m-1})$.

1. Найти самый старший ненулевой бит последовательности $X = x_0 x_1 x_2 x_3 ....$

2. Начальная итерация $i \leftarrow 0$;

$$X_0 \leftarrow Y_0 \leftarrow Z_0 \leftarrow 1 .$$

3. $i \leftarrow i + 1$.

4. Найти X_i на основе x_i:

$$\begin{cases} x_i = 0: & X_i \leftarrow X_{i-1} \cdot 2. \\ x_i = 1: & X_i \leftarrow X_{i-1} \cdot 2 + x_i . \end{cases}$$

5. Найти Y_i на основе x_i:

$$\begin{cases} x_i = 0: & Y_i \leftarrow Y_{i-1} \cdot 4. \\ x_i = 1: & Y_i \leftarrow Y_{i-1} \cdot 4 + 2 \cdot X_i - 1. \end{cases}$$

6. Найти Z_i на основе x_i:

$$\begin{cases} x_i = 0: & Z_i \leftarrow Z_{i-1} \cdot 8. \\ x_i = 1: & Z_i \leftarrow Z_{i-1} \cdot 8 + 3 \cdot Y_i - 3 \cdot X_i + 1. \end{cases}$$

7. Если x_i не последний бит, то перейти к шагу 3.

Особенностью данного алгоритма является то, что параллельно с вычислением куба числа находим также квадрат числа (табл. 3).

Таблица 3. Пример выполнения Алгоритма 3 при вычислении куба числа $11_{10} = 1011_2$

Шаг	Операция
1	Старший бит ненулевой
2	$i \leftarrow 0;\ X_0 \leftarrow Y_0 \leftarrow Z_0 \leftarrow 1$
3	$i \leftarrow 1$
4;5;6	Так как $x_1 = 0$, то $X_1 \leftarrow 1 \cdot 2 = 2_{10} = 10_2$; $Y_1 \leftarrow 1 \cdot 4 = 4_{10} = 100_2$; $Z_1 \leftarrow 1 \cdot 8 = 8_{10} = 1000_2$
7	Бит x_1 не последний, перейти к шагу 3
3	$i \leftarrow 2$
4;5;6	Так как $x_2 = 1$, то $X_2 \leftarrow 2 \cdot 2 + 1 = 5_{10} = 101_2$; $Y_2 \leftarrow 4 \cdot 4 + 2 \cdot 5 - 1 = 25_{10} = 11001_2$; $Z_2 \leftarrow 8 \cdot 8 + (3 \cdot 25 - 3 \cdot 5 + 1) =$ $= 125_{10} = 1111101_2$
7	Бит x_2 не последний, перейти к шагу 3
3	$i \leftarrow 3$
4;5;6	Так как $x_3 = 1$, то $X_3 \leftarrow 5 \cdot 2 + 1 = 11_{10} = 1011_2$; $Y_3 \leftarrow 25 \cdot 4 + 2 \cdot 11 - 1 = 121_{10} =$ $= 1111001_2$; $Z_3 \leftarrow 125 \cdot 8 + (3 \cdot 121 - 3 \cdot 11 + 1) =$ $= 1331_{10} = 10100110011_2$
7	Бит x_3 последний, завершить работу алгоритма

Сравнивая Алгоритм 1 (табл. 1) и Алгоритм 3 (табл. 3) можно заметить, что каждая итерация i Алгоритма 3 имеет на один шаг больше.

Алгоритм 4. Пошаговое описание алгоритма вычисления кубического корня.

ВХОД: Z – битовая последовательность, m – длина последовательности Z ($m \geq 4$).

ВЫХОД: $X = \lfloor \sqrt[3]{Z} \rfloor$, $Y = X^2$,

$$R = Z - X^3 \ (X_i = \lfloor \sqrt[3]{Z_i} \rfloor,\ X_i = \lfloor \sqrt[3]{Y_i} \rfloor,$$

$$R_i = Z_i - X_i^3,\ i = \overline{0, \left\lfloor \frac{m-1}{3} \right\rfloor}).$$

R_i – промежуточная битовая последовательность длины в $3(i+1)$ бит на каждой итерации.

1. Если длина битовой последовательности Z не делится на 3 нацело, то добавляются старшие нулевые биты слева. Разбить на тройки бит $Z = z_0 z_1 z_2 \mid z_3 z_4 z_5 \mid z_6 z_7 z_8 \mid \ldots$.

2. Найти самую старшую ненулевую битовую тройку.

3. Начальная итерация $i \leftarrow 0;\ x_0 \leftarrow X_0 \leftarrow 1,$ $Y_0 \leftarrow Z_0 \leftarrow 1,\ R_0 \leftarrow z_0 z_1 z_2;\ R_0 \leftarrow R_0 - Z_0$.

4. $i \leftarrow i + 1$.

5. $X_i \leftarrow X_{i-1} \cdot 2 + 1$.

6. $dY_i \leftarrow 2 \cdot X_i - 1;\ Y_i \leftarrow Y_{i-1} \cdot 4 + dY_i$.

7. $dZ_i \leftarrow 3 \cdot Y_i - 3 \cdot X_i + 1$.

8. $R_i \leftarrow R_{i-1} \cdot 8 + z_{3i} z_{3i+1} z_{3i+2}$.

9. Если $R_i > dZ_i$, то $x_i \leftarrow 1,\ R_i \leftarrow R_i - dZ_i$, иначе $x_i \leftarrow 0,\ Y_i \leftarrow Y_i - dY_i$ (см. рис. 3, 4).

10. Если биты $z_{3i} z_{3i+1} z_{3i+2}$ не последние, то перейти к шагу 4.

По окончании алгоритма X_i будет содержать целую часть от кубического корня, а R_i – остаток.

Таблица 4. Пример выполнения Алгоритма 4 при вычислении кубического корня числа $1353_{10} = 10101001001_2 = (11_{10})^3 + 2 \cdot 11_{10}$

Шаг	Операции
1	2
1	Добавить старший нулевой бит слева и разбить $Z = 010 \mid 101 \mid 001 \mid 001$ на тройки бит
2	Самая старшая битовая тройка ненулевая
3	Начальная итерация $i \leftarrow 0$; $x_0 \leftarrow 1,\ X_0 \leftarrow Y_0 \leftarrow Z_0 \leftarrow 1$; $R_0 \leftarrow 010_2 - 1 = 1$

Окончание табл. 4

1	2
4	$i \leftarrow 1$
5	$X_1 \leftarrow 1 \cdot 2 + 1 = 3_{10} = 11_2$
6	$dY_1 \leftarrow 2 \cdot 3 - 1 = 5_{10} = 101_2$; $Y_1 \leftarrow 1 \cdot 4 + 5 = 9_2 = 1001_2$
7	$dZ_1 \leftarrow 3 \cdot 9 - 3 \cdot 5 + 1 = 13_{10} = 1101_2$
8	$R_1 \leftarrow 1 \cdot 8 + 5 = 13_{10} = 1101_2$
9	Так как $13 = 13$, то $x_1 \leftarrow 0$ ($X_1 = 2$), $Y_1 \leftarrow 9 - 5 = 4_{10} = 100_2$
10	Биты $z_3 z_4 z_5$ не последние, перейти к шагу 4
4	$i \leftarrow 2$
5	$X_2 \leftarrow 2 \cdot 2 + 1 = 5_{10} = 101_2$
6	$dY_2 \leftarrow 2 \cdot 5 - 1 = 9_{10} = 1001_2$; $Y_2 \leftarrow 4 \cdot 4 + 9 = 25_2 = 11001_2$
7	$dZ_2 \leftarrow 3 \cdot 25 - 3 \cdot 5 + 1 = 61_{10} = 111101_2$
8	$R_2 \leftarrow 13 \cdot 8 + 1 = 105_{10} = 1101000_2$
9	Так как $105 > 61$, то $x_2 \leftarrow 1$, $R_2 \leftarrow 105 - 61 = 44_{10} = 101100_2$
10	Биты $z_6 z_7 z_8$ не последние, перейти к шагу 4
4	$i \leftarrow 3$
5	$X_3 \leftarrow 5 \cdot 2 + 1 = 11_{10} = 1011_2$
6	$dY_3 \leftarrow 2 \cdot 11 - 1 = 21_{10} = 10101_2$; $Y_3 \leftarrow 25 \cdot 4 + 21 = 121_2 = 1111001_2$
7	$dZ_3 \leftarrow 3 \cdot 121 - 3 \cdot 11 + 1 =$ $= 331_{10} = 101001011_2$
8	$R_3 \leftarrow 44 \cdot 8 + 1 = 353_{10} = 101100001_2$
9	Так как $353 > 331$, то $x_3 \leftarrow 1$, $R_3 \leftarrow 353 - 331 = 22_{10} = 10110_2$
10	Биты $z_9 z_{10} z_{11}$ последние, завершить работу алгоритма

Сравнивая Алгоритм 2 (табл. 2) и Алгоритм 4 (табл. 4) можно заметить, что каждая итерация i Алгоритма 4 имеет на один шаг больше.

Возведение в степень n и вычисление корней степени n

Возведение до степени n. Найдем сначала общую итерационную формулу для вычисления степени числа выше третей.

Теорема 1. Выражение $D^n = (A+B)^n$ вычисляется на основе значений меньших степеней D^{n-1}, D^{n-2}, ... , D^2, D^1 и имеет вид:

$$D^n = (A+B)^n = A^n + B \cdot \sum_{i=1}^{n} (-1)^{i-1} \cdot C_n^i \cdot D^{n-i} =$$

$$= A^n + \begin{pmatrix} C_n^1 \cdot D^{n-1} - C_n^2 \cdot D^{n-2} + \ldots + \\ + (-1)^{n-2} \cdot C_n^{n-1} \cdot D + (-1)^{n-1} \end{pmatrix} \cdot B,$$

где A, B, D, n – целые числа, $A, B, D, n > 0$, $C_n^i = \dfrac{n!}{i! \cdot (n-i)!}$, $B^n = B^{n-1} = \ldots$ $\ldots = B^2 = B$.

Доказательство. Найдем сначала выражения для $C_{n-1}^i + C_{n-1}^{i+1}$, $i = \overline{0, n-2}$, в общем виде:

$$C_{n-1}^i + C_{n-1}^{i+1} =$$

$$= \frac{(n-1)!}{i! \cdot (n-1-i)!} + \frac{(n-1)!}{(i+1)! \cdot (n-1-(i+1))!} =$$

$$= \frac{(i+1) \cdot (n-1)!}{(i+1) \cdot i! \cdot (n-1-i)!} + \frac{(n-1-i) \cdot (n-1)!}{(n-1-i) \cdot (i+1)! \cdot (n-i)!} =$$

$$= \frac{(i+1) \cdot (n-1)!}{(i+1) \cdot i! \cdot (n-1-i)!} + \frac{(n-1-i) \cdot (n-1)!}{(i+1)! \cdot (n-1-i)!} =$$

$$= \frac{((i+1) + (n-1-i)) \cdot (n-1)!}{(i+1)! \cdot (n-1-i)!} =$$

$$= \frac{n \cdot (n-1)!}{(i+1)! \cdot (n-(1+i))!} =$$

$$= \frac{n!}{(i+1)! \cdot (n-(1+i))!} = C_n^{i+1}.$$

Выражение вида $C_{n-1}^i + C_{n-1}^{i+1} = C_n^{i+1}$, $i = \overline{0, n-1}$, соответствует значениям треугольника Паскаля для строк $n-1$ и n.

Найдем выражение для 1-й степени:

$$D = A + B.$$

Найдем выражение для 2-й степени:

$$D^2 = (A+B)^2 = A^2 + 2AB + B^2 =$$
$$= A^2 + (2A+1)\cdot B.$$

Сделаем замену A на $D-B$:

$$A^2 + (2D-2B+1)\cdot B = A^2 + (2D-1)\cdot B.$$

Найдем выражение для 3-й степени:

$$D^3 = D^2 \cdot (A+B) =$$
$$= (A^2 + (2D-1)\cdot B)\cdot (A+B) =$$
$$= A^3 + (2D-1)\cdot B\cdot A + (A^2 + 2D-1)\cdot B.$$

Сделаем замену A на $D-B$:

$$A^3 + (2D-1)\cdot B\cdot (D-B) +$$
$$+ (D-B)^2 \cdot B + (2D-1)\cdot B =$$
$$= A^3 + (2D-1)\cdot B\cdot D + (D-B)^2 \cdot B =$$
$$= A^3 + (2D^2 - D)\cdot B + (D^2 - 2D + 1)\cdot B =$$
$$= A^3 + (3D^2 - 3D + 1)\cdot B.$$

Используем методом математической индукции. Найдем выражение n-й степени на основании значения $n-1$-й степени:

$$D^n = (A+B)\cdot D^{n-1} = (A+B)\cdot$$
$$\cdot \left(A^{n-1} + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-2} - C_{n-1}^2 \cdot D^{n-3} + \ldots \\ + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D + (-1)^{n-2} \end{pmatrix} \cdot B \right) =$$
$$= A^n + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-2} - C_{n-1}^2 \cdot D^{n-3} \cdot + \ldots + \\ + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D + (-1)^{n-2} \end{pmatrix} \cdot B \cdot A +$$
$$+ \left(A^{n-1} + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-2} - C_{n-1}^2 \cdot D^{n-3} + \ldots + \\ + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D + (-1)^{n-2} \end{pmatrix} \right) \cdot B.$$

Сделаем замену A на $D-B$:

$$A^n +$$
$$+ \begin{pmatrix} C_{n-1}^1 \cdot D^{n-2} - C_{n-1}^2 \cdot D^{n-3} + \\ + .. + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D + \\ + (-1)^{n-2} \end{pmatrix} \cdot B \cdot (D-B) +$$
$$+ A^{n-1} \cdot B + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-2} - C_{n-1}^2 \cdot D^{n-3} + \\ + \ldots + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D + \\ + (-1)^{n-2} \end{pmatrix} \cdot B =$$
$$= A^n + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-1} - C_{n-1}^2 \cdot D^{n-2} + \\ + .. + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D^2 + \\ + (-1)^{n-2} \cdot D \end{pmatrix} \cdot B +$$
$$+ A^{n-1} \cdot B.$$

С учетом того, что $(D-B)^{n-1}\cdot B$ равносильно $(D-1)^{n-1}\cdot B$ ($B^{n-1} = \ldots = B^2 = B$ по условию), сделаем замену A^{n-1} на $(D-1)^{n-1}$ и воспользуемся формулой Ньютона

$$(D-1)^n = \sum_{i=0}^{n} (-1)^i \cdot C_n^i \cdot D^{n-i},\ C_n^i = \frac{n!}{i!\cdot(n-i)!}:$$
$$A^n + \begin{pmatrix} C_{n-1}^1 \cdot D^{n-1} - C_{n-1}^2 \cdot D^{n-2} \cdot + \ldots + \\ + (-1)^{n-3} \cdot C_{n-1}^{n-2} \cdot D^2 + (-1)^{n-2} \cdot D \end{pmatrix} \cdot B +$$
$$+ \begin{pmatrix} D^{n-1} - C_{n-1}^1 D^{n-2} + \ldots + \\ + (-1)^{n-2} \cdot C_{n-1}^{n-2} \cdot D + (-1)^{n-1} \end{pmatrix} \cdot B.$$

Сгруппируем по степени D:

$$A^n + \begin{pmatrix} (1 + C_{n-1}^1) \cdot D^{n-1} - \\ - (C_{n-1}^1 + C_{n-1}^2) \cdot D^{n-2} \cdot + \ldots + \\ + (-1)^{n-3} \cdot (C_{n-1}^{n-3} + C_{n-1}^{n-2}) \cdot D^2 + \\ + (-1)^{n-2} (C_{n-1}^{n-2} + 1) \cdot D + \\ + (-1)^{n-1} \end{pmatrix} \cdot B.$$

С учетом выражения $C_{n-1}^i +$ $+ C_{n-1}^{i+1} = C_n^{i+1}$, $i = \overline{0, n-1}$ окончательно получаем:

$$A^n + \begin{pmatrix} C_n^1 \cdot D^{n-1} - C_n^2 \cdot D^{n-2} \cdot + \ldots + \\ + (-1)^{n-3} \cdot C_n^{n-2} \cdot D^2 + \\ + (-1)^{n-2} C_n^{n-1} \cdot D + (-1)^{n-1} \end{pmatrix} \cdot B.$$

Теорема доказана.

Рассмотрим полученные выражения:

$$D = A + B,$$
$$D^2 = A^2 + (2D-1)\cdot B,$$
$$D^3 = A^3 + (3D^2 - 3D + 1)\cdot B.$$

Выражения для больших степеней находятся аналогично, используя треугольник Паскаля (см. рис. 5):

n														
0							1							
1						1		1						
2					1		2		1					
3				1		3		3		1				
4			1		4		6		4		1			
5		1		5		10		10		5		1		
6		1		6		15		20		15		6		1
7	1		7		21		35		35		21		7		1

Рис. 5. Вычисление треугольника Паскаля $C_{n,k}$, $k=\overline{0,n}$, $n=\overline{0,7}$

$$D^4 = A^4 + (4D^3 - 6D^2 + 4D - 1)\cdot B,$$

$$D^5 = A^5 + (5D^4 - 10D^3 + 10D^2 - 5D + 1)\cdot B,$$

$$D^6 = A^6 + \begin{pmatrix} 6D^5 - 15D^4 + 20D^3 - \\ -15D^2 + 6D - 1 \end{pmatrix}\cdot B,$$

$$D^7 = A^7 + \begin{pmatrix} 7D^6 - 21D^5 + 35D^4 - \\ -35D^3 + 21D^2 - 7D + 1 \end{pmatrix}\cdot B.$$

Алгоритм 5. Пошаговое описание алгоритма вычисления степени n числа X (табл. 5).

ВХОД: X – битовая последовательность, n – степень, m – длина последовательности X ($m \ge 1$).

ВЫХОД: $D_k = X^k$, $k=\overline{1,n}$.

1. Найти самый старший ненулевой бит последовательности $X = x_0x_1x_2x_3\ldots$.

2. Начальная итерация $i \leftarrow 0$: Для k от 1 до n.

3. $D_k \leftarrow 1$.

4. Конец цикла по k.

5. Для i от 1 до $m-1$

6. $b \leftarrow 1$.

7. Для j от 1 до n

8. $b \leftarrow b\cdot 2;\ D_j \leftarrow D_j \cdot b$.

9. Если $x_i \ne 0$, то

10. $sign \leftarrow 1,\ C \leftarrow j$.

11. Для k от $j-1$ до 1 шаг -1

Таблица 5. Пример выполнения Алгоритма 5 при вычислении 4-й степени числа $5_{10} = 101_2$

Шаг	Операция
1	2
1	Старший бит ненулевой
2;3;4	$i \leftarrow 0$; $D_1 \leftarrow D_2 \leftarrow D_3 \leftarrow D_4 \leftarrow 1$
5;6;7	$i \leftarrow 1;\ b \leftarrow 1;\ j \leftarrow 1$
8	$b \leftarrow 1\cdot 2 = 2;\ D_1 \leftarrow 1\cdot 2 = 2$
9	Так как $x_2 = 0$, то пропустить шаги 10–15
16;17;7	$j \leftarrow 2$
8	$b \leftarrow 2\cdot 2 = 4_{10} = 100_2$; $D_3 \leftarrow 1\cdot 4 = 4_{10} = 100_2$
9	Так как $x_2 = 0$, то пропустить шаги 10–15
16;17;7	$j \leftarrow 3$
8	$b \leftarrow 4\cdot 2 = 8_{10} = 1000_2$; $D_3 \leftarrow 1\cdot 8 = 8_{10} = 1000_2$
9	Так как $x_2 = 0$, то пропустить шаги 10–15
16;17;7	$j \leftarrow 4$
8	$b \leftarrow 8\cdot 2 = 16_{10} = 10000_2$; $D_4 \leftarrow 1\cdot 16 = 16_{10} = 1000_2$

Продолжение табл. 5

1	2
9;16;17	Так как $x_2 = 0$, то пропустить шаги 10–15
5;6;7	$i \leftarrow 2;\ b \leftarrow 1;\ j \leftarrow 1$
8	$b \leftarrow 1 \cdot 2 = 2;$ $D_1 \leftarrow 2 \cdot 2 = 4_{10} = 100_2$
9	Так как $x_3 \neq 0$, то продолжить следующий шаг 10
10	$sign \leftarrow 1,\ C \leftarrow 1$
11	Так как $k = 1 - 1 = 0 < 1$, то пропустить шаги 12–14
15	$D_1 \leftarrow 4 + 1 \cdot 1 = 5_{10} = 101_2$
16;17;7	$j \leftarrow 2$
8	$b \leftarrow 2 \cdot 2 = 4_{10} = 100_2;$ $D_2 \leftarrow 4 \cdot 4 = 16_{10} = 10000_2$
9	Так как $x_3 \neq 0$, то продолжить следующий шаг 10
10	$sign \leftarrow 1,\ C \leftarrow 2$
11	Так как $k = 2 - 1 = 1 \geq 1$, то продолжить шаг 12
12	$D_2 \leftarrow 16 + 1 \cdot 2 \cdot 5 = 26_{10} = 11010_2;$ $C \leftarrow \frac{2 \cdot 1}{(2 - 1 + 1)} = 1$
13	$sign \leftarrow -1$
14;15	$D_2 \leftarrow 26 - 1 \cdot 1 = 25_{10} = 11001_2$
16;17;7	$j \leftarrow 3$
8	$b \leftarrow 4 \cdot 2 = 8_{10} = 1000_2;$ $D_3 \leftarrow 8 \cdot 8 = 64_{10} = 1000000_2$
9	Так как $x_3 \neq 0$, то продолжить следующий шаг 10
10	$sign \leftarrow 1,\ C \leftarrow 3$
11	Так как $k = 3 - 1 = 2 > 1$, то продолжить шаг 12
12	$D_3 \leftarrow 64 + 1 \cdot 3 \cdot 25 = 139_{10} =$ $= 10001011_2;\ C \leftarrow \frac{3 \cdot 2}{(3 - 2 + 1)} = 3$
13	$sign \leftarrow -1$
11	Так как $k = 1$, то продолжить шаг 12

Окончание табл. 5

1	2
12	$D_3 \leftarrow 139 - 1 \cdot 3 \cdot 5 = 124_{10} =$ $= 1111100_2;$ $C \leftarrow \frac{3 \cdot 1}{(3 - 1 + 1)} = 1$
13	$sign \leftarrow 1$
14;15	$D_3 \leftarrow 124 + 1 \cdot 1 = 125_{10} = 1111101_2$
16;17;7	$j \leftarrow 4$
8	$b \leftarrow 8 \cdot 2 = 16_{10} = 10000_2;$ $D_4 \leftarrow 16 \cdot 16 = 256_{10} = 100000000_2$
9	Так как $x_3 \neq 0$, то продолжить следующий шаг 10
10	$sign \leftarrow 1,\ C \leftarrow 4$
11	Так как $k = 4 - 1 = 3 > 1$, то продолжить шаг 12
12	$D_4 \leftarrow 256 + 1 \cdot 4 \cdot 125 =$ $= 756_{10} = 1011110100_2;$ $C \leftarrow \frac{4 \cdot 3}{(4 - 3 + 1)} = 6$
13	$sign \leftarrow -1$
11	Так как $k = 2 > 1$,то продолжить шаг 12
12	$D_4 \leftarrow 756 - 1 \cdot 6 \cdot 25 =$ $= 606_{10} = 1001011110_2;$ $C \leftarrow \frac{6 \cdot 2}{(4 - 2 + 1)} = 4$
13	$sign \leftarrow 1$
11	Так как $k = 1 \geq 1$,то продолжить шаг 12
12	$D_4 \leftarrow 606 + 1 \cdot 4 \cdot 5 =$ $= 626_{10} = 1001110010_2;$ $C \leftarrow \frac{4 \cdot 1}{(4 - 1 + 1)} = 1$
13	$sign \leftarrow 1$
14;15	$D_4 \leftarrow 626 - 1 \cdot 1 = 625_{10} =$ $= 1001110011_2$
16;17;18	Завершить работу алгоритма

12. $D_j \leftarrow D_j + sign \cdot C \cdot D_k$;

$$C \leftarrow \frac{C \cdot k}{(j-k+1)}.$$

13. $sign \leftarrow -sign$.

14. Конец цикла по k.

15. $D_j \leftarrow D_j + sign \cdot 1$.

16. Конец если.

17. Конец цикла по j.

18. Конец цикла по i.

По завершении алгоритма $D_1 = 5$, $D_2 = 25$, $D_3 = 125$, $D_4 = 625$.

Оптимизация за счет исключения операций деления. Значения коэффициента C – небольшие числа в табл. 5. Если значения C вычислить раньше, то можно исключить часть операций умножения и все операции деления в табл. 5. На Шаге 12 Алгоритма 5 выражение $C \leftarrow \frac{C \cdot k}{(j-k+1)}$ можно избежать, используя коэффициент $C_{j,k}$ в выражении $D_j \leftarrow D_j + sign \cdot C_{j,k} \cdot D_k$, где j, k – номера строки и элемента строки в треугольнике Паскаля (см. рис. 5), что исключает операции деления в Алгоритме 5.

Далее в табл. 6 показаны наибольшие значения коэффициентов в каждой строке треугольника Паскаля (см. рис. 5). С учетом того, что $2^{32} = 4294967296$, можно утверждать, что $C_{n,\left\lceil \frac{n+1}{2} \right\rceil} < 2^{32}$ при $n \le 34$ в табл. 6. Возведение в степень n числа вычисляется на основании меньших степеней $n-1$, $n-2$, … 2, 1, 0, то при возведении в степень $n \le 35$ умножение на $C_{j,k+1}$ в Алгоритме 5 является однословной операцией, что значительно уменьшает сложность по числу однословных операций. Из табл. 6 также видно, что при возведении в степень до $n \le 5$ значения коэффициентов очень маленькие: 1, 2, 3, 4, 5, 6, …. В этом случае операций умножения можно избежать полностью, заменив их несколькими дополнительными операциями многословного сложения.

Анализ сложности по числу битовых операций. Довольно непросто найти общую сложность Алгоритма 5, так как присутствуют битовые операции сдвига, побитового сложения и вычитания, умножения однословного числа на битовую последовательность. Попробуем вычислить число битовых операций и операций умножения на однословное число.

Лемма 3. Число операций однословного умножения в Алгоритме 5 можно выразить следующей формулой $(n-1) \cdot \frac{n \cdot m}{2}$, где m – длина в битах последовательности, которая возводится в степень n.

Таблица 6. Наибольшее значение $C_{n,\left\lceil \frac{n+1}{2} \right\rceil}$ в каждой строке треугольника Паскаля (рис. 5)

n	Значение	n	Значение	n	Значение	n	Значение
0	1	10	252	20	184756	30	155117520
1	1	11	462	21	352716	31	300540195
2	2	12	924	22	705432	32	601080390
3	3	13	1716	23	1352078	33	1166803110
4	6	14	3432	24	2704156	34	2333606220
5	10	15	6435	25	5200300	35	4537567650
6	20	16	12870	26	10400600	36	9075135300
7	35	17	24310	27	20058300		
8	70	18	48620	28	40116600		
9	126	19	92378	29	77558760		

Доказательство. Строки $n = 0, 1$ содержат единицы, которые заменяются операцией сложения, поэтому эти строки не рассматриваются. Анализируя Алгоритм 5 и треугольник Паскаля, очевидно, что коэффициенты со строки $n = 2$ треугольника Паскаля при возведении в степень 4 например, будут использоваться три раза. С учетом того, что первый и последний коэффициенты – единицы, следующее выражение описывает число однословных операций при возведении в 2, 3, 4, 5-ю степени:

2-я степень: $1 \cdot m = 1m$;

3-я степень: $(1+2) \cdot m = 3m$;

4-я степень: $(1+2+3) \cdot m = 6m$;

5-я степень: $(1+2+3+4) \cdot m = 10m$;

6-я степень: $(1+2+3+4+5) \cdot m = 15m$;

где m – длина битовой последовательности.

Коэффициенты при m описываются формулой $(n-1) \cdot \frac{n}{2}$ для арифметической прогрессии. Умножив на m, получаем искомое выражение.

Лемма доказана.

Лемма 4. Число операций битового сложения и вычитания в Алгоритме 5 можно выразить следующей формулой:

$$(n-1) \cdot \frac{n}{2} \cdot m \cdot \frac{(m+1)}{2} + m,$$

где m – длина в битах последовательности, которая возводится в степень n.

Доказательство. Битовая последовательность длины m при возведении в степень n будет иметь длину nj бит. С учетом того, что значение D^n может быть получены на основе значений меньших степеней D^{n-1}, D^{n-2}, …, D, 1 (согласно теоремы 1) и, рассматривая шаги 11–14, число бытовых операций сложения и вычитания можно выразить формулой для арифметической прогрессии:

$$(n-1) \cdot \frac{n}{2} \cdot m + 1.$$

Рассмотрим число операций для различных длин m:

$$m = 1: \ (n-1) \cdot \frac{n}{2} \cdot 1 + 1.$$

$$m = 2: \ (n-1) \cdot \frac{n}{2} \cdot 2 + 1.$$

….

$$m-1: \ (n-1) \cdot \frac{n}{2} \cdot (m-1) + 1.$$

$$m: \ (n-1) \cdot \frac{n}{2} \cdot m + 1.$$

Не учитывая второе слагаемое +1, видим, что значения формируют также арифметическую прогрессию с постоянным коэффициентом $(n-1) \cdot \frac{n}{2}$. Сумма таких значений может быть выражена следующей формулой:

$$(n-1) \cdot \frac{n}{2} \cdot m \cdot \frac{(m+1)}{2} + m.$$

Лемма доказана.

Вычисление корня n-й степени.

Алгоритм 6. Пошаговое описание алгоритма вычисления корня n-й степени числа Q (табл. 7).

ВХОД: Q – битовая последовательность, n – степень $(n > 0)$, m – длина последовательности Q $(m \geq n)$.

ВЫХОД: $D_1 = \left\lfloor \sqrt[n]{Q} \right\rfloor$; $D_k = (D_1)^k$, $k = \overline{2, n}$; $R = Q - (D_1)^n$ – остаток.

R_i – промежуточная битовая последовательность длиной в $n(i+1)$ бит на каждой итерации i.

1. Если длина последовательности Q не делится на n нацело, то добавляются $m - \left\lfloor \frac{m}{n} \right\rfloor \cdot n$ старших нулевых бит слева. Разбить $Q = q_0 q_1 \ldots q_{n-1} \mid q_n q_{n+1} \ldots q_{2n-1} \mid \ldots$ на секции длиной в n бит.

2. Найти самую старшую ненулевую битовую секцию.

3. Начальная итерация $i \leftarrow 0$:Для k от 1 до n.

4. $D_k \leftarrow 1$.

5. Конец цикла по k.

6. $R_0 \leftarrow q_0 q_1 ... q_{n-1}$; $R_0 \leftarrow R_0 - D_n$.

7. Для i от 1 до $\left\lceil \frac{m-1}{n} \right\rceil$.

8. $b \leftarrow 2$; $D_1 \leftarrow D_1 \cdot b$; $T_1 \leftarrow D_1 + 1$.

9. Для j от 2 до n

10. $b \leftarrow b \cdot 2$; $D_j \leftarrow D_j \cdot b$; $T_j \leftarrow D_j$.

11. $sign \leftarrow 1$.

12. Для k от $j-1$ до 1 шаг -1

13. $T_j \leftarrow T_j + sign \cdot C_{j,k} \cdot T_k$.

14. $sign \leftarrow -sign$.

15. Конец цикла по k.

16. $T_j \leftarrow T_j + sign \cdot 1$.

17. Конец цикла по j.

18. $R_i \leftarrow R_{i-1} \cdot b + q_{ni} q_{ni+1} .... q_{ni+n-1}$.

19. Если $R_i > T_n$, то

20. $R_i \leftarrow R_i - T_n$; $T_n \leftarrow 0$.

21. Для j от 1 до n

22. $D_j \leftarrow T_j$.

23. Конец цикла по j.

24. Конец если.

25. Конец цикла по i.

Таблица 7. Пример выполнения Алгоритма 6 при вычислении корня 4-ой степени числа $640_{10} = 1010000000_2 = (5_{10})^4 + 3 \cdot 5_{10}$, $m = 10$, $n = 4$

Шаг	Операция
1	2
1	Длина последовательности 1010000000_2 не кратна четырем. Добавить два старших нулевых бита 001010000000_2
2; 3; 4	$i \leftarrow 0$; $D_1 \leftarrow D_2 \leftarrow D_3 \leftarrow D_4 \leftarrow 1$
5; 6	$R_0 \leftarrow 0010_2 = 2_{10}$; $R_0 \leftarrow 2 - 1 = 1$
7	$i \leftarrow 1$
8	$b \leftarrow 2$; $D_1 \leftarrow 1 \cdot 2$; $T_1 \leftarrow 2 + 1 = 3_{10} = 11_2$
9	$j \leftarrow 2$

Продолжение табл. 7

1	2
10	$b \leftarrow 2 \cdot 2 = 4_{10} = 100_2$; $D_2 \leftarrow 1 \cdot 4 = 4_{10} = 100_2$; $T_2 \leftarrow 4$
11	$sign \leftarrow 1$
12	$k \leftarrow 1$
13	$T_2 \leftarrow 4 + 1 \cdot 2 \cdot 3 = 10_{10} = 1010_2$ $(C_{2,2} = 2)$
14	$sign \leftarrow -1$
15; 16	$T_2 \leftarrow 10 - 1 \cdot 1 = 9_{10} = 1001_2$
17; 9	$j \leftarrow 3$
10	$b \leftarrow 4 \cdot 2 = 8_{10} = 1000_2$; $D_3 \leftarrow 1 \cdot 8 = 8_{10} = 1000_2$; $T_3 \leftarrow 8$
11	$sign \leftarrow 1$
12	$k \leftarrow 2$
13	$T_3 \leftarrow 8 + 1 \cdot 3 \cdot 9 = 35_{10} = 100011_2$ $(C_{3,3} = 3)$
14	$sign \leftarrow -1$
12	$k \leftarrow 1$
13	$T_3 \leftarrow 35 - 1 \cdot 3 \cdot 3 = 26_{10} = 11010_2$ $(C_{3,2} = 3)$
14	$sign \leftarrow 1$
15;16	$T_3 \leftarrow 26 + 1 \cdot 1 = 27_{10} = 11011_2$
17; 9	$j \leftarrow 4$
10	$b \leftarrow 8 \cdot 2 = 16_{10} = 10000_2$; $D_4 \leftarrow 1 \cdot 16 = 16_{10} = 10000_2$; $T_4 \leftarrow 16$
11	$sign \leftarrow 1$
12	$k \leftarrow 3$
13	$T_4 \leftarrow 16 + 1 \cdot 4 \cdot 27 = 124_{10} =$ $= 1111100_2$ $(C_{4,4} = 4)$
14	$sign \leftarrow -1$
12	$k \leftarrow 2$
13	$T_4 \leftarrow 124 - 1 \cdot 6 \cdot 9 = 70_{10} =$ $= 1000110_2$ $(C_{4,3} = 6)$
14	$sign \leftarrow 1$
12	$k \leftarrow 1$
13	$T_4 \leftarrow 70 + 1 \cdot 4 \cdot 3 = 82_{10} = 1010010_2$ $(C_{4,2} = 4)$

Продолжение табл. 7

1	2
14	$sign \leftarrow -1$
15; 16	$T_4 \leftarrow 82-1\cdot 1=81_{10}=1010001_2$
17; 18	$R_2=2\cdot 16+1000_2=40_{10}=11000_2$
19	Так как $40<81$, то пропустить шаги 20, 21, 22, 23, 24
25; 7	$i \leftarrow 2$
8	$b \leftarrow 2$; $D_1 \leftarrow 2\cdot 2$; $T_1 \leftarrow 4+1=5_{10}=101_2$
9	$j \leftarrow 2$
10	$b \leftarrow 2\cdot 2=4_{10}=100_2$; $D_2 \leftarrow 4\cdot 4=16_{10}=10000_2$; $T_2 \leftarrow 16$
11	$sign \leftarrow 1$
12	$k \leftarrow 1$
13	$T_2 \leftarrow 16+1\cdot 2\cdot 5=26_{10}=11010_2$ $(C_{2,2}=2)$
14	$sign \leftarrow -1$
15; 16	$T_2 \leftarrow 26-1\cdot 1=25_{10}=11001_2$
17; 9	$j \leftarrow 3$
10	$b \leftarrow 4\cdot 2=8_{10}=1000_2$ $D_3 \leftarrow 8\cdot 8=64_{10}=1000000_2$; $T_3 \leftarrow 64$
11	$sign \leftarrow 1$
12	$k \leftarrow 2$
13	$T_3 \leftarrow 64+1\cdot 3\cdot 25=$ $=139_{10}=10001011_2$ $(C_{3,3}=3)$
14	$sign \leftarrow -1$
12	$k \leftarrow 1$
13	$T_3 \leftarrow 139-1\cdot 3\cdot 5=$ $=124_{10}=1111100_2$ $(C_{3,2}=3)$
14	$sign \leftarrow 1$
15; 16	$T_3 \leftarrow 124+1\cdot 1=125_{10}=1111101_2$
17; 9	$j \leftarrow 4$
10	$b \leftarrow 8\cdot 2=16_{10}=10000_2$; $D_4 \leftarrow 16\cdot 16=256_{10}=$

Окончание табл. 7

1	2
	$=100000000_2$; $T_4 \leftarrow 256$
11	$sign \leftarrow 1$
12	$k \leftarrow 3$
13	$T_4 \leftarrow 256+1\cdot 4\cdot 125=$ $=756_{10}=1011110100_2$ $(C_{4,4}=4)$
14	$sign \leftarrow -1$
12	$k \leftarrow 2$
13	$T_4 \leftarrow 756-1\cdot 6\cdot 25=$ $=606_{10}=1001011110_2$ $(C_{4,3}=6)$
14	$sign \leftarrow 1$
12	$k \leftarrow 1$
13	$T_4 \leftarrow 606+1\cdot 4\cdot 5=$ $=626_{10}=1001110010_2$ $(C_{4,2}=4)$
14	$sign \leftarrow -1$
15; 16	$T_4 \leftarrow 626-1\cdot 1=$ $=625_{10}=1001110001_2$
17; 18	$R_2=40\cdot 16+0=$ $=640_{10}=1010000000_2$
19	Так как $640>625$, то продолжить шаг 20
20	$R_2 \leftarrow 640-625=15_{10}=1111_2$; $T_4 \leftarrow 0$
21;22; 23	$D_1 \leftarrow 5$, $D_2 \leftarrow 25$, $D_3 \leftarrow 125$, $D_4 \leftarrow 625$
24; 25	Завершить алгоритм

Результат вычисления корня 4-й степени равен $D_1=5_{10}$, остаток $R_2=15_{10}$.

Лемма 3 применима также к Алгоритму 6. Число однословных операций умножения для вычисления квадратного и кубического корней соответствует выражениям m и $3m$, это говорит о том, что сложность вычисления данных корней по числу таких операций имеет линейную сложность.

Вывод

В работе описаны метод возведения в степень n (Алгоритм 5) и метод вычисления корня степени n (Алгоритм 6) больших чисел на основе рекуррентных соотношений ($n>1$). В Теореме 1 доказано, что методы возведения в степени 2 и 3 являются частными случаями обобщенного метода возведения в степень n большого числа. Проведен анализ сложности по числу битовых операций сложения, вычитания и умножения алгоритма возведения в степень n в зависимости от длины в битах m исходного большого числа и степени n. Сравнивая алгоритмы, видно, что Алгоритм 6 вычисления корня степени n больших чисел основан на Алгоритме 5 вычисления степени n с небольшими изменениями. Показано, каким образом исключаются операции деления в предложенных алгоритмах. Спецификой описанных алгоритмов является то, что при возведении большого числа X в степень n или вычислении корня степени n одновременно с нахождением результата вычисляются значения всех степеней до $n-1$, что можно эффективно использовать в «Гибком алгоритме возведения в степень по модулю» для предварительного вычисления степеней. Методы по своей природе являются побитовыми, поэтому рекомендуется использование методов для вычисления степеней, не превышающих 35 ($n \le 35$), так как в этом случае длина в битах коэффициентов треугольника Паскаля не превышает длины машинного слова в 32 бита, что позволяет минимизировать использование многословных операций. В работе приведены схема возведения в квадрат и схема вычисления квадратного корня. В работе доказаны 4 леммы. В виде таблиц продемонстрированы примеры выполнения описанных алгоритмов для различных степеней.

1. *Кнут Д.* Искусство программирования для ЭВМ: Пер. с англ. / Под ред. Бабенко. – М.: Мир, 1977. – 2. – 734 с.
2. *Задірака В., Олексюк О.* Ком'пютерна арифметика багаторозрядних чисел. – К., Наук. видання. – 2003. – 263 с.
3. *Карацуба А.А., Офман Ю.П.* Умножение многоразрядных чисел на автоматах // ДАН СССР, 145 (1962). – С. 293–294.
4. *Березовский А.И., Задирака В.К., Шевчук Л.Б.* О тестировании быстродействия алгоритмов и программ выполнения основных операций для ассиметричной криптографии // Кибернетика и системный анализ. – 1999. – № 5. – С. 61–68.
5. *Терещенко А.Н.* Быстрое вычисление квадратного и кубического корней без использования операций умножения и деления // Искусственный интеллект. – 2006. – № 3. – С. 783–792.

Получено 12.09.2014

Об авторе:

Терещенко Андрей Николаевич,
кандидат физико-математических наук.

Место работы автора:

ООО "СимКорп - Украина"
Старший инженер-программист
Киев, Стусса 35/37.
E-mail: teramidi@ukr.net