КІБЕРНЕТИКА та КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

УДК 519.6 DOI:10.34229/2707-451X.25.1.4

А.М. ТЕРЕЩЕНКО

ПІДНЕСЕННЯ ДО КВАДРАТУ БАГАТОРОЗРЯДНОГО ЧИСЛА НА 8 ПРОЦЕСОРАХ У ПАРАЛЕЛЬНІЙ МОДЕЛІ ОБЧИСЛЕНЬ

Розглядається операція піднесення до квадрату N-розрядного числа, яка є однією з основних операцій шифрування, дешифрування, верифікації ключів асиметричної криптографії, та від швидкодії якої залежить швидкодія асиметричних криптографічних програмно-апаратних комплексів. Запропоновано алгоритм реалізації операції піднесення до квадрату N-розрядного числа ($N = 2^n$, $N \geq 4$) для обчислення якого необхідно виконати вісім операцій множення чисел довжиною N/4 розрядів (три з яких є операціями піднесення до квадрату), що на одне множення менше ніж у методі Карацуби, який виконується рекурсивно. Запропонований алгоритм зменшує кількість операцій множення за рахунок спрощення структури числа, яка досягається завдяки штучній повторюваності частин числа. Наведено приклад обчислення операції піднесення до квадрату 4-х розрядного числа на основі запропонованого методу. Запропонований метод може бути використано рекурсивно, є зручним для розпаралелювання у разі реалізації на багатопроцесорних системах.

Ключові слова: *багаторозрядна арифметика, багаторозрядне множення, багаторозрядне піднесення до квадрату, багаторозрядне піднесення до квадрату за модулем, асиметрична криптографія, паралельна модель обчислень.*

© А.М. Терещенко, 2025

Вступ. На даний час безпека та конфіденційність даних стає нагальним викликом не тільки для приватних осіб, а і на державному рівні. Рівень кібератак виріс на стільки, що хакери можуть виводити з ладу сервери, змінювати дані у хмарних сховищах, що унеможливлює роботу цілих державних органів на тривалий період, що потребує значних зусиль на відновлення ресурсів. Кібератаки стають більш організованими та вибірковими. Використання асиметричних криптографічних програмно-апаратних комплексів [1–3] стає обов'язковим елементом захисту. Швидкодія асиметричних криптографічних операцій таких, як шифрування, дешифрування, верифікації ключів, залежить від швидкодії операції піднесення до квадрату за модулем. Використовуючи швидкі методи, операція піднесення до квадрату *N*-розрядного числа може бути виконана з меншою складністю ніж операція множення двох *N*-розрядних чисел.

До 1962 року вважалося, що оцінка складності операції множення N^2 – оптимальна, але публікація А. Карацуби [4] показала, що множення дворозрядних чисел може бути виконано з використанням трьох множень замість чотирьох. Метод Карацуби був поштовхом для пошуку нових швидких алгоритмів реалізації багаторозрядної операції множення, які значно прискорювали виконання алгоритмів, спрощували реалізацію на мікропроцесорах та заощаджували використання енергії [5–11]. Різні швидкі алгоритми багаторозрядного множення та піднесення до квадрату мають різні діапазони ефективного використання, що звужує їх ефективне використання та оптимізацію тільки у цих діапазонах. Тому для подальшого прискорення багаторозрядного піднесення до квадрату у більшому діапазоні чисел існує нагальна потреба у розробці алгоритмів, виконання яких може бути розпаралелено.

У даній роботі розглядається метод реалізації операції піднесення до квадрату, який дозволяє зменшити кількість операцій множення та піднесення до квадрату. Метод є зручним для розпаралелювання, що збільшує його діапазон ефективного використання.

Постановка задачі. Нехай 4-розрядне число представлене у вигляді

$$U_4 = u_0 + u_1 \cdot W + u_2 \cdot W^2 + u_3 \cdot W^3 \ , \ W = 2^{\omega},$$

де ω – довжина машинного розряду у бітах. Тоді операція піднесення до квадрату може бути представлена у наступному вигляді:

$$(U_4)^2 = \left(\sum_{i=0}^{3} u_i \cdot W^i \right)^2 = \sum_{i=0}^{3} (u_i^2 \cdot W^{2i}) + 2 \sum_{i=0}^{2} \sum_{j=i+1}^{3} (u_i \cdot u_j \cdot W^{i+j}) \ , \ W = 2^{\omega}, \qquad (1)$$

де ω – довжина машинного розряду у бітах.

Для обчислення (1) необхідно виконати чотири однорозрядні операції піднесення до квадрату та шість однорозрядних операцій множення. Загалом необхідно виконати 10 операції множення. Операцію множення на число 2 не враховуємо, так як така операція може бути замінена операцією додавання.

Використовуючи метод Карацуби рекурсивно, обчислення операції піднесення до квадрату 4-розрядного числа можна виконати за 9 однорозрядних операцій множення.

Необхідно побудувати алгоритм обчислення операції піднесення до квадрату 4-розрядного числа, який використовує 8 однорозрядних операцій множення, три з яких це операції піднесення до квадрату.

Необхідно отримати оцінку складності розробленого методу піднесення до квадрату *N*-розрядного числа. Отриману оцінку складності використати для порівняння розробленого методу з методом Карацуби.

Піднесення до квадрату 4-розрядного числа. Розглянемо теорему.

Теорема 1. Обчислення операції піднесення числа $U_4 = u_0 + u_1 \cdot W + u_2 \cdot W^2 + u_3 \cdot W^3$ до квадрату, де $0 \le u_i < W$, $N = 2^n$, $n \ge 2$, $W = 2^{\omega}$, ω – довжина машинного розряду у бітах, потребує три однорозрядні операції піднесення до квадрату та п’ять однорозрядних операцій множення.

Доведення. Введемо додаткові числа a_0, a_1, b_0, b_1, щоб отримати простішу структуру числа U_4 за рахунок повторюваності частин числа. Для цього представимо число U_4 у вигляді вектора-стовпця:

$$U_4 = \begin{bmatrix} u_3 \cdot W^3 \\ u_2 \cdot W^2 \\ u_1 \cdot W^1 \\ u_0 \cdot W^0 \end{bmatrix} = \begin{bmatrix} a_1 & \cdot W^3 \\ (a_1 + b_1) & \cdot W^2 \\ (a_0 + b_0) & \cdot W^1 \\ a_0 & \cdot W^0 \end{bmatrix}, \ W = 2^{\omega}, \qquad (2)$$

де ω – довжина машинного розряду у бітах; $0 \le a_0, a_1 < W$; $-W < b_0, b_1 < W$; $b_0 = u_1 - a_0$, $b_1 = u_2 - a_1$, $a_0 = u_0$, $a_1 = u_3$. Таке представлення у вигляді вектора-стовпця не використовується для векторних операцій, а тільки для полегшення візуального сприйняття виразів.

Числа b_0 та b_1 можуть бути від’ємними, що потребує додаткових операцій для врахування знаків переносу або запозичення. Але такі операції реалізуються за рахунок додаткових операцій додавання та віднімання, що не впливає на загальну кількість операцій множення і у даній теоремі не розглядаються.

Піднесення до квадрату (2) можна представити у вигляді вектора-стовпця:

$$\left(\begin{bmatrix} a_1 & \cdot W^3 \\ (a_1+b_1) & \cdot W^2 \\ (a_0+b_0) & W^1 \\ a_0 & \cdot W^0 \end{bmatrix}\right)^2 = \left(\begin{bmatrix} a_1 \cdot W \\ a_1 \end{bmatrix} \cdot W^2 + \begin{bmatrix} b_1 \cdot W \\ b_0 \end{bmatrix} \cdot W + \begin{bmatrix} a_0 \cdot W \\ a_0 \end{bmatrix}\right)^2.$$

Після розкриття дужок отримуємо вираз:

$$\left(\begin{bmatrix} a_1 \cdot W \\ a_1 \end{bmatrix} \cdot W^2\right)^2 + \left(\begin{bmatrix} a_0 \cdot W \\ a_0 \end{bmatrix}\right)^2 + \left(\begin{bmatrix} b_1 \cdot W \\ b_0 \end{bmatrix} \cdot W\right)^2 +$$

$$+2 \cdot \begin{bmatrix} a_1 \cdot W \\ a_1 \end{bmatrix} \cdot \begin{bmatrix} a_0 \cdot W \\ a_0 \end{bmatrix} \cdot W^2 + 2 \cdot \begin{bmatrix} a_1 \cdot W \\ a_1 \end{bmatrix} \cdot \begin{bmatrix} b_1 \cdot W \\ b_0 \end{bmatrix} \cdot W^3 + 2 \cdot \begin{bmatrix} a_0 \cdot W \\ a_0 \end{bmatrix} \cdot \begin{bmatrix} b_1 \cdot W \\ b_0 \end{bmatrix} \cdot W.$$

Продовжуючи розкриття дужок, отримуємо наступний вираз:

$$\begin{gathered}
(a_1^2 \cdot W^2 + 2 \cdot a_1^2 \cdot W + a_1^2) \cdot W^4 + \\
+(a_0^2 \cdot W^2 + 2 \cdot a_0^2 \cdot W + a_0^2) + \\
+(b_1^2 \cdot W^2 + 2 \cdot b_1 \cdot b_0 \cdot W + b_0^2) \cdot W^2 + \\
+2 \cdot (a_1 \cdot a_0 \cdot W^2 + 2 \cdot a_1 \cdot a_0 \cdot W + a_1 \cdot a_0) \cdot W^2 + \\
+2 \cdot (a_1 \cdot b_1 \cdot W^2 + a_1 \cdot (b_0 + b_1) \cdot W + a_1 \cdot b_0) \cdot W^3 + \\
+2 \cdot (a_0 \cdot b_1 \cdot W^2 + a_0 \cdot (b_0 + b_1) \cdot W + a_0 \cdot b_0) \cdot W.
\end{gathered}$$

Розкриємо дужки для обчислення степенів *W*:

$$\begin{gathered}
a_1^2 \cdot W^6 + 2 \cdot a_1^2 \cdot W^5 + a_1^2 \cdot W^4 + \\
+a_0^2 \cdot W^2 + 2 \cdot a_0^2 \cdot W + a_0^2 + \\
+b_1^2 \cdot W^4 + 2 \cdot b_1 \cdot b_0 \cdot W^3 + b_0^2 \cdot W^2 + \\
+2 \cdot a_1 \cdot a_0 \cdot W^4 + 4 \cdot a_1 \cdot a_0 \cdot W^3 + 2 \cdot a_1 \cdot a_0 \cdot W^2 + \\
+2 \cdot a_1 \cdot b_1 \cdot W^5 + 2 \cdot a_1 \cdot (b_0 + b_1) \cdot W^4 + 2 \cdot a_1 \cdot b_0 \cdot W^3 + \\
+2 \cdot a_0 \cdot b_1 \cdot W^3 + 2 \cdot a_0 \cdot (b_0 + b_1) \cdot W^2 + 2 \cdot a_0 \cdot b_0 \cdot W.
\end{gathered}$$

Згрупуємо доданки за степенями *W*:

$$\begin{gathered}
a_1^2 \cdot W^6 + 2 \cdot a_1 \cdot (a_1 + b_1) \cdot W^5 + \\
+(a_1^2 + b_1^2 + 2 \cdot a_1 \cdot a_0 + 2 \cdot a_1 \cdot (b_0 + b_1)) \cdot W^4 + \\
+(2 \cdot b_1 \cdot b_0 + 4 \cdot a_1 \cdot a_0 + 2 \cdot a_1 \cdot b_0 + 2 \cdot a_0 \cdot b_1) \cdot W^3 + \\
+(a_0^2 + b_0^2 + 2 \cdot a_1 \cdot a_0 + 2 \cdot a_0 \cdot (b_0 + b_1)) \cdot W^2 + \\
+2 \cdot a_0 \cdot (a_0 + b_0) \cdot W + a_0^2.
\end{gathered}$$

Зробимо заміни $m_0 = (a_0)^2$, $m_6 = (a_1)^2$, $m_1 = 2 \cdot a_0 \cdot (a_0 + b_0)$, $m_5 = 2 \cdot a_1 \cdot (a_1 + b_1)$ та отримаємо наступний вираз:

$$m_6 \cdot W^6 + m_5 \cdot W^5 +$$
$$+(m_6 + b_1^2 + 2 \cdot a_1 \cdot a_0 + 2 \cdot a_1 \cdot (b_0 + b_1)) \cdot W^4 +$$
$$+(2 \cdot b_1 \cdot b_0 + 4 \cdot a_1 \cdot a_0 + 2 \cdot a_1 \cdot b_0 + 2 \cdot a_0 \cdot b_1) \cdot W^3 +$$
$$+(m_0 + b_0^2 + 2 \cdot a_1 \cdot a_0 + 2 \cdot a_0 \cdot (b_0 + b_1)) \cdot W^2 +$$
$$+m_1 \cdot W + m_0.$$

Використаємо наступні вирази $m_7 = 2 \cdot (a_0 + b_0) \cdot (a_1 + b_1)$, $m_3 = (a_0 + a_1)^2$ у дужках для W^2, W^3, W^4 та отримаємо наступне:

$$m_6 \cdot W^6 + m_5 \cdot W^5 +$$
$$+(m_6 + b_1^2 + m_7 - 2 \cdot b_0 \cdot b_1 - 2 \cdot a_0 \cdot b_1 + 2 \cdot a_1 \cdot b_1) \cdot W^4 +$$
$$+(m_3 - m_0 - m_6 + m_7) \cdot W^3 +$$
$$+(m_0 + b_0^2 + m_7 - 2 \cdot b_0 \cdot b_1 - 2 \cdot b_0 \cdot a_1 + 2 \cdot b_0 \cdot a_0) \cdot W^2 +$$
$$+m_1 \cdot W + m_0.$$

Введемо вирази $m_2 = b_0 \cdot (2 \cdot a_0 + b_0 - 2 \cdot a_1 - 2 \cdot b_1)$, $m_4 = b_1 \cdot (2 \cdot a_0 + b_0 - 2 \cdot a_0 - 2 \cdot b_0)$ та остаточно отримаємо вирази:

$$m_6 \cdot W^6 + m_5 \cdot W^5 +$$
$$+(m_6 + m_4 + m_7) \cdot W^4 +$$
$$+(m_3 - m_0 - m_6 + m_7) \cdot W^3 +$$
$$+(m_0 + m_2 + m_7) \cdot W^2 +$$
$$+m_1 \cdot W + m_0, \qquad (3)$$

де

$$m_0 = (a_0)^2; \quad m_1 = 2 \cdot a_0 \cdot (a_0 + b_0);$$
$$m_6 = (a_1)^2; \quad m_5 = 2 \cdot a_1 \cdot (a_1 + b_1);$$
$$m_2 = b_0 \cdot (2 \cdot a_0 + b_0 - 2 \cdot a_1 - 2 \cdot b_1);$$
$$m_4 = b_1 \cdot (2 \cdot a_1 + b_1 - 2 \cdot a_0 - 2 \cdot b_0);$$
$$m_3 = (a_0 + a_1)^2; \quad m_7 = 2 \cdot (a_0 + b_0) \cdot (a_1 + b_1). \qquad (4)$$

У виразі (3) використовуються значення m_i, $i = \overline{0,7}$, отримані з використанням восьми операцій множення (4), три з яких $m_0 = (a_0)^2$, $m_6 = (a_1)^2$, $m_3 = (a_0 + a_1)^2$ це операції піднесення до квадрату. У формулі (3) операції множення на 2 можуть бути замінені операціями додавання.

Теорему доведено.

Алгоритм піднесення до квадрату 4-х розрядного числа.

На основі виразів (3), (4) можна побудувати алгоритм піднесення до квадрату.

Алгоритм 1. Піднесення до квадрату 4-х розрядного числа.

Вхідні дані: U_4 – 4-розрядне число, B – основа числення.

Результат: $R_8 \leftarrow (U_4)^2$.

1. Підготовка даних:

$$a_0 \leftarrow u_0;$$
$$a_1 \leftarrow u_3;$$
$$b_0 \leftarrow u_1 - a_0;$$
$$b_1 \leftarrow u_2 - a_1.$$

2. Множення:

$$m_0 \leftarrow (a_0)^2;$$
$$m_6 \leftarrow (a_1)^2;$$
$$m_1 \leftarrow 2 \cdot a_0 \cdot (a_0 + b_0);$$
$$m_5 \leftarrow 2 \cdot a_1 \cdot (a_1 + b_1);$$
$$m_2 \leftarrow b_0 \cdot (2 \cdot a_0 + b_0 - 2 \cdot a_1 - 2 \cdot b_1);$$
$$m_4 \leftarrow b_1 \cdot (2 \cdot a_1 + b_1 - 2 \cdot a_0 - 2 \cdot b_0);$$
$$m_3 \leftarrow (a_0 + a_1)^2;$$
$$m_7 \leftarrow 2 \cdot (a_0 + b_0) \cdot (a_1 + b_1).$$

3. Обчислення елементів результату:

$$r_0 \leftarrow m_0;$$
$$r_1 \leftarrow m_1;$$
$$r_2 \leftarrow m_0 + m_2 + m_7;$$
$$r_3 \leftarrow m_3 - m_0 - m_6 + m_7;$$
$$r_4 \leftarrow m_6 + m_4 + m_7;$$
$$r_5 \leftarrow m_5;$$
$$r_6 \leftarrow m_6.$$

4. Формування результату:

$$R_8 \leftarrow (r_0, r_1, r_2, r_3, r_4, r_5, r_6, 0).$$

5. Корегування результату:

$$r_i \leftarrow \langle r_i \rangle_B,\ r_{i+1} \leftarrow r_{i+1} + \lfloor r_i / B \rfloor, i = \overline{0,6}.$$

Позначення в алгоритмі.

$a_0 \leftarrow u_0$ – операція «стрілка вліво» відрізняється від операції «=» та використовується в алгоритмі для позначення обчислення виразу з правого боку та занесенням результату в комірку, яка відповідає елементу зліва. Приклад: $m_0 \leftarrow (a_0)^2$.

Порядок виконання операцій визначається символами «,» та «;». Символ «;» ділить операції на групи операцій. Групи операцій виконуються зліва направо. Символ «,» ділить операції всередині групи операцій. Операції, розділені символом «,», виконуються справа наліво на відміну від операцій, розділених символом «;». У наведеному прикладі 4, 3, 2, 1; 6, 5; 10, 9, 8 нумерація відповідає порядковому номеру виконання операції. У наступному прикладі $r_i \leftarrow \langle r_i \rangle_B,\ r_{i+1} \leftarrow r_{i+1} + \lfloor r_i / B \rfloor,$

$i=\overline{0,6}$ зміна значення елементу $r_i \leftarrow \langle r_i \rangle_B$ виконується в останню чергу, спочатку значення r_i використовується для обчислення r_{i+1}, що відбувається у циклі $i=\overline{0,6}$.

Приклади обчислення алгоритму піднесення до квадрату 4-х розрядного числа.

Приклад 2. Різнорідна структура числа [12].

Обчислення операції піднесення до квадрату числа 1235_{10} відповідно до Алгоритму 1.

Вхідні дані: $U_4 \leftarrow (5,3,2,1)$, B=10.

$$u_0=5,\ u_1=3,\ u_2=2,\ u_3=1.$$

1. Підготовка даних:

$$a_0 \leftarrow u_0 = 5;$$
$$a_1 \leftarrow u_3 = 1;$$
$$b_0 \leftarrow u_1 - a_0 = 3-5 = -2;$$
$$b_1 \leftarrow u_2 - a_1 = 2-1 = 1.$$

2. Множення:

$$m_0 \leftarrow (a_0)^2 = 5\cdot 5 = 25;$$
$$m_6 \leftarrow (a_1)^2 = 1\cdot 1 = 1;$$
$$m_1 \leftarrow 2\cdot a_0\cdot(a_0+b_0) = 2\cdot 5\cdot(5-2) = 30;$$
$$m_5 \leftarrow 2\cdot a_1\cdot(a_1+b_1) = 2\cdot 1\cdot(1+1) = 4;$$
$$m_2 \leftarrow b_0\cdot(2\cdot a_0+b_0-2\cdot a_1-2\cdot b_1) = -2\cdot(2\cdot 5-2-2\cdot 1-2\cdot 1) = -8;$$
$$m_4 \leftarrow b_1\cdot(2\cdot a_1+b_1-2\cdot a_0-2\cdot b_0) = 1\cdot(2\cdot 1+1-2\cdot 5-2\cdot(-2)) = -3;$$
$$m_3 \leftarrow (a_0+a_1)^2 = (5+1)^2 = 36;$$
$$m_7 \leftarrow 2\cdot(a_0+b_0)\cdot(a_1+b_1) = 2\cdot(5-2)\cdot(1+1) = 12.$$

3. Обчислення елементів результату:

$$r_0 \leftarrow m_0 = 25;$$
$$r_1 \leftarrow m_1 = 30;$$
$$r_2 \leftarrow m_0+m_2+m_7 = 25-8+12 = 29;$$
$$r_3 \leftarrow m_3-m_0-m_6+m_7 = 36-25-1+12 = 22;$$
$$r_4 \leftarrow m_6+m_4+m_7 = 1-3+12 = 10;$$
$$r_5 \leftarrow m_5 = 4;$$
$$r_6 \leftarrow m_6 = 1.$$

4. Формування результату:

$$R_8 \leftarrow (25,30,29,22,10,4,1,0).$$

5. Корегування результату:

$$r_0 \leftarrow \langle r_0 \rangle_{10} = \langle 25 \rangle_{10} = 5,\ r_1 \leftarrow r_1 + \lfloor r_0/B \rfloor = 30 + \lfloor 25/10 \rfloor = 32,\ i=0;$$
$$r_1 \leftarrow \langle r_1 \rangle_{10} = \langle 32 \rangle_{10} = 2,\ r_2 \leftarrow r_2 + \lfloor r_1/B \rfloor = 29 + \lfloor 32/10 \rfloor = 32,\ i=1;$$
$$r_2 \leftarrow \langle r_2 \rangle_{10} = \langle 32 \rangle_{10} = 2,\ r_3 \leftarrow r_3 + \lfloor r_2/B \rfloor = 22 + \lfloor 32/10 \rfloor = 25,\ i=2;$$
$$r_3 \leftarrow \langle r_3 \rangle_{10} = \langle 25 \rangle_{10} = 5,\ r_4 \leftarrow r_4 + \lfloor r_3/B \rfloor = 10 + \lfloor 25 \rfloor = 12,\ i=3;$$

$$r_4 \leftarrow \langle r_4 \rangle_{10} = \langle 12 \rangle_{10} = 2,\ r_5 \leftarrow r_5 + \lfloor r_4 / B \rfloor = 4 + \lfloor 12 \rfloor = 5,\ i = 4;$$
$$r_5 \leftarrow \langle r_5 \rangle_{10} = \langle 5 \rangle_{10} = 5,\ r_6 \leftarrow r_6 + \lfloor r_5 / B \rfloor = 1 + \lfloor 5 \rfloor = 1,\ i = 5;$$
$$r_6 \leftarrow \langle r_6 \rangle_{10} = \langle 1 \rangle_{10} = 1,\ r_7 \leftarrow r_7 + \lfloor r_6 / B \rfloor = 0 + \lfloor 1 \rfloor = 0,\ i = 6.$$

Отримуємо:

$$R_8 \leftarrow (5,2,2,5,2,5,1,0).$$

Перевірка результату $1235_{10} \cdot 1235_{10} = 01525225_{10}$.

Складність операції піднесення до квадрату *N*-розрядного числа за кількістю однорозрядних множень. Розглянемо наступну теорему.

Теорема 2. Загальна кількість однорозрядних операцій множення у вищенаведеному алгоритму піднесення до квадрату *N*-розрядного числа ($N = 2^n$, $n \geq 2$) може бути оцінена наступними виразами рекурсивно:

$$O_N^2 = 3 \cdot O_{N/4}^2 + 5 \cdot O_{N/4}^*,$$
$$O_N^* = 3 \cdot O_{N/2}^*,$$
$$O_2^2 = O_2^* = 3,\ O_1^2 = O_1^* = 1, \qquad (5)$$

де O_N^2, $O_{N/4}^2$, O_2^2, O_1^2 – оцінки складності операції піднесення до квадрату чисел довжиною *N*, *N*/4 розрядів, 2 розряди та 1 розряд запропонованим методом; O_N^*, $O_{N/2}^*$, O_2^*, O_1^* – оцінки складності множення двох чисел довжиною *N*, *N*/2 розрядів, 2 розряди та 1 розряд методом Карацуби.

Доведення. Представимо число U_N у вигляді вектора-стовпця:

$$U_N = \begin{bmatrix} V_3 \cdot P^3 \\ V_2 \cdot P^2 \\ V_1 \cdot P^1 \\ V_0 \cdot P^0 \end{bmatrix} = \begin{bmatrix} A_1 & \cdot P^3 \\ (A_1 + B_1) & \cdot P^2 \\ (A_0 + B_0) & \cdot P^1 \\ A_0 & \cdot P^0 \end{bmatrix},$$
$$V_j = \sum_{i=0}^{N/4-1} (u_{i+N/4 \cdot j} \cdot W^i),\ j = \overline{0,3},$$

де $P = W^{N/4}$, $W = 2^{\omega}$, ω – довжина машинного розряду у бітах; $0 \leq A_0, A_1 < P$; $-P < B_0, B_1 < P$; $B_0 = V_1 - A_1$, $B_1 = V_2 - A_1$, $A_0 = V_0$, $A_1 = V_3$.

Як доведено у теоремі 1 піднесення до квадрату числа U_4 потребує виконання наступних операцій:

$m_0 = (a_0)^2$; $\quad m_1 = 2 \cdot a_0 \cdot (a_0 + b_0)$; $m_2 = b_0 \cdot (2 \cdot a_0 + b_0 - 2 \cdot a_1 - 2 \cdot b_1)$;

$m_6 = (a_1)^2$; $\quad m_5 = 2 \cdot a_1 \cdot (a_1 + b_1)$; $m_4 = b_1 \cdot (2 \cdot a_1 + b_1 - 2 \cdot a_0 - 2 \cdot b_0)$;

$m_3 = (a_0 + a_1)^2$; $\quad m_7 = 2 \cdot (a_0 + b_0) \cdot (a_1 + b_1)$.

Для обчислення m_0, m_3, m_6 необхідно виконати три операції піднесення до квадрату чисел довжиною *N*/4 розрядів. Для обчислення m_1, m_2, m_4, m_5, m_7 необхідно виконати п’ять операцій множення чисел довжиною *N*/4 розрядів методом Карацуби.

Для обчислення m_0, m_3, m_6 можна використати теорему 1 рекурсивно. Для піднесення до квадрату кожного з чисел довжиною $N/4$ розрядів потрібно виконати три операції піднесення до квадрату чисел довжиною $N/16$ ($N \geq 16$) розрядів та п'ять операцій множення чисел довжиною $N/16$ ($N \geq 16$) розрядів.

Множення на число 2 у формулі (3) може бути замінено на операції додавання, тому такі операції не враховуємо.

Оцінка складності $O_N^* = 3 \cdot O_{N/2}^*$ методу Карацуби є відомою і не потребує доведення.

Теорему доведено.

Порівняння методів. Використовуючи вирази (5), отримаємо загальну кількість однорозрядних множень, необхідних для реалізації операції піднесення до квадрату чисел довжиною N розрядів. Коефіцієнт прискорення операції піднесення до квадрату запропонованим методом у порівнянні з методом Карацуби в залежності від N наведено у таблиці.

ТАБЛИЦЯ. Порівняння оцінок складності операції піднесення до квадрату запропонованим методом та методом Карацуби чисел довжиною N розрядів за кількістю однорозрядних множень

$N = 2^n$	Піднесення до квадрату, $O_N^2 = 3 \cdot O_{N/4}^2 + 5 \cdot O_{N/4}^*$, для $N \geq 4$	Метод Карацуби, $O_N^* = 3^n$	O_N^2 / O_N^*
1	1	1	1
2	3	3	1
4	8	9	0.8888
8	24 = 3 · 3 + 5 · 3	27	0.8888
16	69 = 3 · 8 + 5 · 9	81	0.8519
32	207 = 3 · 24 + 5 · 27	243	0.8519
64	612 = 3 · 69 + 5 · 81	729	0,8395
128	1836 = 3 · 207 + 5 · 243	2187	0,8395
256	5481 = 3 · 612 + 5 · 729	6561	0,8354
512	16443 = 3 · 1836 + 5 · 2187	19683	0,8354
1024	49248 = 3 · 5481 + 5 · 6561	59049	0,8340

З табл. 1 видно, що для $N \geq 4$ коефіцієнт прискорення буде від 11 % до 16 % зі збільшенням N.

Висновки. У роботі запропоновано метод реалізації операції піднесення до квадрату числа довжиною N розрядів ($N = 2^n$, $N \geq 4$) з можливістю розпаралелення. Показано, що для 4-х розрядного числа операція піднесення до квадрату може бути виконана з використанням восьми однорозрядних операцій множення, що на одне множення менше ніж у методі Карацуби, який використовується рекурсивно. Для 4-х розрядного числа розроблений метод потребує три однорозрядні операції піднесення до квадрату та п'ять однорозрядних операцій множення. У роботі наведено алгоритм реалізації операції піднесення до квадрату 4-х розрядного числа та надано приклад обчислення алгоритму. Проведена оцінка складності запропонованого методу для чисел довжиною N розрядів та показано, що обчислення може бути виконано на основі трьох піднесень до квадрату чисел довжиною $N/4$ розрядів та п'яти множень чисел довжиною $N/4$ розрядів. Запропонований метод може бути використано рекурсивно, що підвищує можливості з розпаралелювання реалізації операції піднесення до квадрату великих чисел. Проведено порівняльний аналіз запропонованого методу та методу Карацуби за кількістю однорозрядних множень в залежності від N. Порівняльний аналіз показав, що для $N \geq 4$ коефіцієнт прискорення складає від 11 % до 16 % зі збільшенням довжини числа. Для перевірки результату обчислення реалізовано у вигляді алгоритму-програми SquaringNumber2aa на

мові програмування APL. Результати роботи можуть бути використані для розробки швидких алгоритмів багаторозрядної арифметики та для реалізації операції піднесення до квадрату великих чисел у мікросхемному виконанні.

Список літератури

1. Rivest R.L., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*. 1978. Vol. 21, N 2. P. 120–126.
2. Задирака В.К., Кудин А.М. Построение программно-аппаратных комплексов арифметики сверхбольших чисел. *Компьютерная математика*. 2001. Т.1. С. 158–163.
3. Tereshchenko A., Zadiraka V. Algorithm for calculation the carry and borrow signs in multi-digit operations in the parallel computational model. *International Journal of Computing*. 2023. **22**(1). P. 21–28. https://doi.org/10.47839/ijc.22.1.2875
4. Карацуба А.А., Офман Ю.П. Умножение многоразрядных чисел на автоматах. *ДАН СССР*. 1962. **145** (2). С. 293–294. http://mi.mathnet.ru/dan26729
5. Cook S.A. On the Minimum Computation Time of Functions, 1966 PhD thesis, Harvard University Department of Mathematics. http://cr.yp.to/bib/1966/cook.html (звернення: 10.02.2025)
6. Schonhage A., Strassen V. Schnelle Multiplikation großen Zahlen. *Computing*. 1971. Vol. 7, No. 3–4. P. 281–292. https://www.scribd.com/doc/68857222/Schnelle-Multiplikation-gro%C3%9Fer-Zahlen
7. Анісімов А.В. Алгоритмічна теорія великих чисел. Модулярна арифметика великих чисел. Київ: Видавничий дім "Академперіодика", 2001. 153 с. http://books.zntu.edu.ua/book_info.pl?id=21106
8. Задірака В., Олексюк О. Комп'ютерна арифметика багаторозрядних чисел: Наукове видання. Київ. 2003. 263 с.
9. Николайчук Я.М., Касянчук М.М., Якименко І.З., Івасьєв С.В. Ефективний метод модулярного множення в теоретико-числовому базисі Радемахера – Крестенсона. *Вісник Національного університету "Львівська політехніка". Комп'ютерні системи та мережі*. 2014. № 806. С. 195–199. http://nbuv.gov.ua/UJRN/VNULPKSM_2014_806_31
10. Хіміч О.М., Сидорук В.А. Використання мішаної розрядності у математичному моделюванні. *Математичне та комп'ютерне моделювання. Серія: Фізико-математичні науки.* 2019. Вип. 19. С. 180–187. http://mcm-math.kpnu.edu.ua/article/view/174244
11. Задірака В.К., Терещенко А.М. Комп'ютерна арифметика багаторозрядних чисел у послідовній та паралельній моделях обчислень. Київ: Наук. думка, 2021. 136 с.
12. Tereshchenko A., Zadiraka V. Generating Big Numbers for Testing Multi-Digit Arithmetic Algorithms. *Cybernetics and Computer Technologies*. 2021. Iss. 2. P. 39–56. https://doi.org/10.34229/2707-451X.21.2.4

Одержано 10.02.2025

Терещенко Андрій Миколайович,
кандидат фізико-математичних наук, докторант
Інституту кібернетики імені В.М. Глушкова НАН України, Київ.
https://orcid.org/0000-0002-9549-9275
teramidi@ukr.net

УДК 519.6

А.М. Терещенко

Піднесення до квадрату багаторозрядного числа на 8 процесорах у паралельній моделі обчислень

Інститут кібернетики імені В.М. Глушкова НАН України, Київ
Листування: teramidi@ukr.net

Вступ. На даний час безпека та конфіденційність даних стає нагальним викликом не тільки для приватних осіб, а і на державному рівні. Рівень кібератак виріс на стільки, що хакери можуть виводи з ладу сервери, змінювати дані в хмарних сховищах, що унеможливлює роботу цілих державних органів на тривалий період, що потребує значних зусиль на відновлення ресурсів. Кібератаки стають більш організованими та вибірковими. Використання асиметричних криптографічних програмно-апаратних комплексів

стає обов'язковим елементом захисту. Швидкодія асиметричних криптографічних операцій таких, як шифрування, дешифрування, верифікації ключів, залежить від швидкодії операції піднесення до квадрату за модулем. Використовуючи швидкі методи, операція піднесення до квадрату числа довжиною N розрядів може бути виконана з меншою складністю ніж операція множення двох чисел довжиною N розрядів.

У даній роботі розглядається метод реалізації операції піднесення до квадрату, який дозволяє зменшити кількість операцій множення та піднесення до квадрату у порівнянні з методом Карацуби. Запропонований метод може бути використано рекурсивно, є зручним для розпаралелювання, що збільшує його діапазон ефективного використання.

Мета роботи. Зменшити кількість однорозрядних операцій множення та піднесення до квадрату для пришвидшення часу виконання операції піднесення до квадрату числа довжиною N розрядів ($N=2^n$). Зменшити кількість множень довжиною $N/4$ розрядів до 8 операцій у разі піднесення до квадрату числа довжиною N розрядів. Зменшити загальну обчислювальну складність та знайти модифікацію, при якій кількість кроків додавання та віднімання буде найменшою у порівнянні з іншими модифікаціями. Побудувати алгоритм піднесення до квадрату на основі знайденої модифікації. Знайти ефективний метод піднесення до квадрату чисел довжиною до 4096 бітів з можливістю розпаралелення.

Результати. Розглянута структура числа, яке розбивається на чотири частини, для реалізації операції піднесення до квадрату. Досліджено вплив розділення 4-х розрядного числа на 2-х розрядне число та 4-х розрядне число з простішою структурою, штучно створеною за рахунок повторюваності його частин, на реалізацію операції піднесення до квадрату. Розроблено метод реалізації операції піднесення до квадрату 4-х розрядного числа за вісім однорозрядних множень, що на одне множення менше ніж у методі Карацуби, який виконується рекурсивно. Для отримання результату операції піднесення до квадрату використовуються лінійні комбінації результатів множення. У роботі наведена така модифікація реалізації, яка має невелику кількість операцій додавань та віднімань у лінійних комбінаціях. Наведено алгоритм на основі запропонованої модифікації, де показано, як 4-х розрядне число з різнорідною структурою може бути розділене на 2-х розрядне число та 4-х розрядне число з простішою структурою за рахунок повторюваності. Проведено аналіз складності за кількістю однорозрядних операцій множення в залежності від довжина числа N у разі рекурсивного використання алгоритму реалізації операції піднесення до квадрату для порівняння з методом Карацуби.

Висновки. У роботі наведено алгоритм реалізації операції піднесення до квадрату 4-х розрядного числа на основі восьми однорозрядних множень, три з яких є операціями піднесення до квадрату. На прикладі показана реалізація операції піднесення до квадрату 4-х розрядного числа з різнорідною структурою. Проведена оцінка складності запропонованого методу для чисел довжиною N розрядів та показано, що обчислення може бути виконано на основі трьох піднесень до квадрату чисел довжиною $N/4$ розрядів та п'яти множень чисел довжиною $N/4$ розрядів. На основі порівняльного аналізу показано, що запропонований метод піднесення до квадрату кращий за метод Карацуби на 11 % для ($N = 4$) та до 16 % зі збільшенням N. Для перевірки результату обчислення реалізовано у вигляді алгоритму-програми SquaringNumber2aa на мові програмування APL. Метод може бути використано рекурсивно, що підвищує можливості з розпаралелювання реалізації операції піднесення до квадрату великих чисел, що збільшує його діапазон ефективного використання до 4096 бітів.

Ключові слова: багаторозрядна арифметика, багаторозрядне множення, багаторозрядне піднесення до квадрату, багаторозрядне піднесення до квадрату за модулем, асиметрична криптографія, паралельна модель обчислень.

MSC 90C15, 49M27

Andrii Tereshchenko

Square a Multi-Digit Number on 8 Processors in a Parallel Computational Model

V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine, Kyiv
Correspondence: teramidi@ukr.net

Introduction. Currently, data security and confidentiality are becoming an urgent challenge not only for individuals, but also at the state level. The level of cyberattacks has grown to such an extent that hackers can disable servers, change data in cloud storage, which makes it impossible to work entire government agencies for a long period, which requires significant efforts to restore resources. Cyberattacks are becoming more organized

and selective. The use of asymmetric cryptographic software and hardware complexes is becoming an obligatory element of protection. The speed of asymmetric cryptographic operations such as encryption, decryption, key verification depends on the speed of the modulo squaring operation. Using fast methods, the operation of squaring a number of N digits can be performed with less complexity than the operation of multiplying two numbers of N digits.

This paper considers a method for implementing the squaring operation, which allows reducing the number of multiplication and squaring operations compared to the Karatsuba method. The proposed method can be used recursively and is convenient for parallelization, which increases its range of effective use.

The purpose of the article is to reduce the number of single-digit multiplication and squaring operations to speed up the execution time of the operation of squaring a number with a length of N digits ($N=2^n$, $N\geq4$). Reduce the number of multiplications with a length of N/4 digits to 8 operations in the case of squaring a number with a length of N digits. Reduce the overall computational complexity and find a modification in which the number of addition and subtraction steps will be the smallest compared to other modifications. Build a squaring algorithm based on the modification found. Find an efficient method of squaring numbers with a length of up to 4096 bits with the possibility of parallelization.

Results. The structure of a number that is divided into four parts for the implementation of the squaring operation is considered. The effect of splitting a 4-digit number into a 2-digit number and a 4-digit number with a simpler structure, artificially created due to the repetition of its parts, on the implementation of the squaring operation is investigated. A method for implementing the squaring operation of a 4-digit number in eight one-digit multiplications is developed, which is one multiplication less than in the Karatsuba method, which is performed recursively. To obtain the result of the squaring operation, linear combinations of the multiplication results are used. The paper presents such a modification of the implementation, which has a small number of addition and subtraction operations in linear combinations. An algorithm based on the proposed modification is presented, which shows how a 4-digit number with a heterogeneous structure can be split into a 2-digit number and a 4-digit number with a simpler structure due to repetition. An analysis of the complexity in terms of the number of single-digit multiplication operations depending on the length of the number N is carried out in the case of recursive use of the algorithm for implementing the square operation for comparison with the Karatsuba method.

Conclusions. The paper presents an algorithm for implementing the operation of squaring a 4-digit number based on eight single-digit multiplications, three of which are squaring operations. The example shows the implementation of the operation of squaring a 4-digit number with a heterogeneous structure. The complexity of the proposed method is estimated for numbers of length N digits, and it is shown that the calculation can be performed on the basis of three squaring operations of numbers of length N/4 digits and five multiplications of numbers of length N/4 digits. Based on the comparative analysis, it is shown that the proposed squaring method is better than the Karatsuba method by 11 % for (N = 4) and by 16 % with increasing N. To verify the result of the calculation, the algorithm-program SquaringNumber2aa in the APL programming language is implemented. The method can be used recursively, which increases the possibilities of parallelizing the implementation of the square operation of large numbers, which increases its effective range of use to 4096 bits.

Keywords: multidigit arithmetic, multidigit multiplication, multidigit squaring, multidigit squaring by modulo, asymmetric cryptography, parallel computational model.