КІБЕРНЕТИКА
та КОМП'ЮТЕРНІ
ТЕХНОЛОГІЇ

УДК 519.6 DOI:10.34229/2707-451X.21.4.7

А.М. ТЕРЕЩЕНКО, В.К. ЗАДІРАКА

РЕАЛІЗАЦІЯ БАГАТОРОЗРЯДНОЇ ОПЕРАЦІЇ МНОЖЕННЯ НА ОСНОВІ ДИСКРЕТНИХ КОСИНУСНИХ ТА СИНУСНИХ ПЕРЕТВОРЕНЬ

Розглядається операція багаторозрядного множення, від швидкодії якої залежить швидкодія асиметричних криптографічних програмно-апаратних комплексів. Запропоновано алгоритми реалізації операції множення двох N-розрядних чисел на основі дискретних косинусних та синусних перетворень (ДКП та ДСП). У запропонованих алгоритмах за рахунок використання ДКП і ДСП розділено обчислення для дійсної та уявної частин дискретного перетворення Фур'є (ДПФ) дійсного сигналу парної довжини, що дозволяє перевести обчислення з поля комплексних чисел у поле дійсних та цілих чисел та зменшити складність багаторозрядної операції множення до лінійної складності за кількістю однорозрядних операцій множення.

Ключові слова: *багаторозрядне множення, багаторозрядна арифметика, асиметрична криптографія, дискретне косинусне перетворення, дискретне синусне перетворення, дискретне перетворення Фур'є, швидкий алгоритм обчислення Фур'є.*

© А.М. Терещенко, В.К. Задірака, 2021

Вступ. Поява нових паралельних обчислювальних систем таких, як багатоядерні процесори, графічні прискорювачі, кластери, розподілені системи, системи з розподіленою пам'яттю та інші обумовлена вирішенням складних прикладних задач у різних галузях. Серед таких задач можна виділити задачі обчислення систем лінійних алгебраїчних рівнянь з кількістю невідомих 33–35 мільйонів, розрахунок оболонок ядерних реакторів, моделювання фізичних, хімічних процесів, аеродинаміки, гідродинаміки, захисту інформації тощо. Це значно розширює використання багаторозрядної арифметики [1 – 7] із-за того, що неврахування похибок заокруглення призводить до того, що іноді отримуються комп'ютерні рішення, які не відповідають фізичному змісту. Багаторозрядні операції множення це складова частина операції піднесення до степеня за модулем, від швидкодії якої залежить швидкодія асиметричних криптографічних програмно-апаратних комплексів.

У роботі Карацуби – Офмана 1962 року [8] надано опис алгоритму, який мав складність обчислення багаторозрядного множення за кількістю однорозрядних множень меншу ніж $O^{Стовпчик}(N^2)$, де N – кількість розрядів (слів) багаторозрядного числа. Історично це був перший алгоритм, який відрізнявся від методу «множення у стовпчик». За своєю природою алгоритм Карацуби – Офмана це рекурсивний алгоритм зі складністю $O^{Карацуби}(N^{\log_2 3})$ меншою ніж для методу «множення у стовпчик», так як на кожному рівні рекурсії обчислювалися три операції множення замість чотирьох операцій. Використання рекурсії для обчислення операції множення була вже відома Чарльзу Беббиджу, винахіднику першої аналітичної обчислювальної машини у 19 сторіччі, але можливість заміни чотирьох множень трьома не була наділена достатньою увагою. З 1962 року почався активний пошук алгоритмів «швидкого множення». У 1965 році Кулі та Такі незалежно один від одного запропонували алгоритм [9], який дозволяв дискретне

перетворення Фур'є довжини $N = N_1 \cdot N_2$ представити у вигляді дискретних перетворень меншої довжини N_2 рекурсивно та зменшити обчислювальну складність до $O^{ШПФ}(N \log N)$. Більш відомою назвою для цього алгоритму є алгоритм «швидкого перетворення Фур'є» або ШПФ. Розробку швидкого алгоритму обчислення дискретного перетворення можна віднести до роботи Гаусса 1805 року, де вінхотів інтерполювати орбіту астероїдів Палас та Юна. Ця робота передує роботам Фур'є, але у роботі Гаусса не була приділена увага обчислювальній складності та зрештою був використаний інший метод. У 1971 році, використовуючи ШПФ, Шенхаге та Штрассен запропонували алгоритм зі складністю $O^{Шенхаге-Штрассен}(N \cdot \log N \cdot \log\log N)$ у кільці з $2^N + 1$ елементів [10]. Цей алгоритм був найшвидшим відомим алгоритмом до 2007 року. Алгоритм Шенхаге – Штрассена фундаментально базується на теоремі про циклічну згортку для перетворення Фур'є, яка свідчить, що результат одновимірної циклічної згортки (наприклад, вимір часу) є результатом множення в іншому виміру (наприклад, виміру частоти). У цьому ж 1971 році Пітассі опублікував роботу, в якій був представлений швидкий алгоритм обчислення перетворення Уошла [11], на основі якого також може бути побудована багаторозрядна операція множення [12]. У 1985 році Монтгомері у своїй статті описав алгоритм, який дозволяє швидко обчислювати операцію множення, якщо вона є складовою операції множення за модулем [13]. З появою багатопроцесорних систем, з'явилась можливість розпаралелити обчислення з лінійною складністю за кількістю однорозрядних операцій множення у межах одного процесора. У своїй дисертаційні роботі 1966 року [14] Кук описав метод Тоома від 1963 року, який за своєю складністю $O^{Тоома}(N^{\log(2k-1)/\log k})$ при великих k наближається до лінійної складності, де k – кількість частин, на які розбивається велике число. При $k = 2$ метод Тоома – Кука співпадає з методом Карацуби – Офмана. У методі Тоома – Кука необхідно обчислювати невідомі та цей метод потребує більшої кількості операцій додавань та віднімань.

Досягнення у обчисленні ДПФ вплинули на розвиток інших дискретних перетворень. Швидкі обчислення дискретних косинусного та синусного перетворень (ДКП та ДСП) почали широко використовуватися у цифровій обробці сигналів починаючи з 1974 та 1976 років відповідно [15 – 19]. У більшості робіт ДКП обчислюється за рахунок використання швидких алгоритмів обчислення дискретного перетворення Фур'є (ДПФ).

У даній роботі, навпаки, для обчислення ДПФ задіяно ДКП та ДСП. У роботі запропоновано метод множення багаторозрядних чисел на основі ДКП та ДСП, у якому розділено обчислення для дійсної та уявної частин ДПФ, що дозволяє перевести обчислення з поля комплексних чисел у поле дійсних та цілих чисел та зменшити складність багаторозрядної операції множення до лінійної складності за кількістю однорозрядних операцій множення.

Постановка задачі. Розглянемо обчислення $R_{2N} = U_N \cdot V_N$, де U_N, V_N – N-розрядні цілі додатні числа, а R_{2N} – $2N$-розрядне число. Такі числа можна представити у вигляді:

$$U_N = (u_{N-1}u_{N-2}...u_0) = \sum_{i=0}^{N-1} u_i 2^{\omega i},\ V_N = (v_{N-1}v_{N-2}...v_0) = \sum_{i=0}^{N-1} v_i 2^{\omega i},\ R_{2N} = (r_{2N-1}r_{2N-2}...r_0) = \sum_{i=0}^{2N-1} r_i 2^{\omega i},$$

де ω – довжина машинного слова у бітах (далі будемо вважати ω=16, 24, 32 чи 64 біта), $0 \le u_i, v_i, r_i < 2^{\omega}$.

Необхідно розробити алгоритм реалізації операції множення двох N-розрядних чисел, який за кількістю операцій однорозрядного множення мав би лінійну складність $O_{con}(N) = kN + C_{con}$ у послідовній моделі обчислення, де k, C_{con} – константи для послідовної моделі обчислень.

Базовий алгоритм множення з обчисленням ДПФ у полі комплексних чисел (ДПФ-2*N*) [2]. В даному алгоритмі до кожного з N-розрядних чисел додається N старших нулів. Отримані $2N$-розрядні числа представляються у вигляді дійсних $2N$-розрядних вектор-стовпців, які можна використовувати як вхідні параметри у ДПФ. Перехід від N – до $2N$-розрядних чисел та використання ДПФ розрядністю $2N$ пояснюється тим, що результатом множення двох N-розрядних чисел буде число розрядністю $2N$.

Алгоритм 1. Множення двох N-розрядних чисел з обчисленням ДПФ розрядністю $2N$ у полі комплексних чисел.

1. Додавання старших нулів: $X_{2N}(N+r) \leftarrow Y_{2N}(N+r) \leftarrow 0$, $r = \overline{0, N-1}$.
2. Обчислення **ДПФ розрядністю** $2N$: $\hat{X}_{2N} \leftarrow W_{2N.2N} \cdot X_{2N}$, $\hat{Y}_{2N} \leftarrow W_{2N,2N} \cdot Y_{2N}$,

де $W_{2N}^{\langle r \cdot k \rangle_{2N}} = e^{-\frac{2\pi \cdot i}{2N} \cdot r \cdot k} = e^{-\frac{\pi \cdot i}{N} \cdot r \cdot k}$, $i = \sqrt{-1}$, $r = \overline{0, 2N-1}$, $k = \overline{0, 2N-1}$, елементи матриці $W_{2N.2N}$.

3. Перемноження **ДПФ розрядністю** $2N$: $\hat{Z}_{2N}(r) \leftarrow \hat{X}_{2N}(r) \cdot \hat{Y}_{2N}(r)$, $r = \overline{0, 2N-1}$.
4. Обчислення **ОДПФ**: $Z_{2N} \leftarrow \frac{1}{2N} \cdot W_{2N.2N} \cdot \hat{Z}_{2N}^{*}$ (або $\frac{1}{2N} \cdot W_{2N,2N}^{*} \cdot \hat{Z}_{2N}$).
5. Обчислення результату: $Z_{2N}(r) \leftarrow [\mathrm{Re}\, Z_{2N}(r)]$, $r = \overline{0, 2N-1}$,

де $[\mathrm{Re}\, Z_{2N}(r)]$ – заокруглення до найближчого цілого дійсної частини $Z_{2N}(r)$.

Теорема 1. Складність Алгоритму 1 за кількістю операцій множення однорозрядних дійсних чисел має наступний вигляд:

$$O_{\mathrm{Algorithm}\,1}^{multiplication}(2N) = 9N \log_2 N + 6N + 9C_{FFT}.$$

Доведення. На кроці 2 для обчислення ДПФ необхідно матрицю $W_{2N,2N}$ розмірами $2N$ на $2N$ комплексних елементів помножити на вектор-стовпець довжиною $2N$ дійсних чисел. Для спрощення порівняння з наступним алгоритмом вважаємо, що множення відбувається як на дійсну, так і на уявну частину елементів матриці $W_{2N,2N}$, навіть якщо дійсна або уявна частина дорівнюють нулю. Складність множення матриці $W_{2N,2N}$ на вектор-стовпець X_{2N} за кількістю операцій множення однорозрядних чисел на комплексні числа з однорозрядними числами у дійсній та уявній частинах буде $O_{DFT}^{multiplication,real}(2N) = (2N)^2$. Використання ШПФ дозволяє зменшити цю оцінку до $O_{FFT}^{multiplication,complex}(2N) = N \log_2 N + C_{FFT}$, якщо вважати, що на першій ітерації «метелика» обчислюються тільки операції додавання та віднімання. Тоді на кроці 2 обчислення двох ДПФ потребує $2N \log_2 N + 2C_{FFT}$ операцій множення комплексних однорозрядних чисел.

На кроці 3 поелементне множення двох вектор-стовбців довжиною $2N$ комплексних чисел потребує $2N$ операцій множення комплексних чисел.

На кроці 4 аналогічно кроку 2 для обчислення ОДПФ необхідно $N \log_2 N + C_{FFT}$ операцій комплексного множення, не враховуючи операції множення на $1/2N$, які можуть бути реалізовані за рахунок бітових зсувів.

Одне множення $u \cdot v = u_{re} v_{re} - u_{im} v_{im} + i(u_{re} v_{im} + u_{im} v_{re})$ двох комплексних чисел $u = u_{re} + i u_{im}$ та $v = v_{re} + i v_{im}$ потребує чотирьох дійсних множень. При використанні значень $A = (u_{re} + u_{im})(v_{re} + v_{im})$, $S = (u_{re} - u_{im})(v_{re} - v_{im})$, комплексне множення може бути обчислене за рахунок трьох операцій множення дійсних чисел наступним чином:

$$\mathrm{Re}(u \cdot v) = A + S - 2u_{im} v_{im}, \ \mathrm{Im}(u \cdot v) = A - S. \quad (1)$$

Операція $-2u_{im}v_{im}$ може бути реалізована за рахунок двох послідовних операцій віднімання, тому її не враховуємо до кількості множень. З урахуванням (1) загальна кількість операцій множення однорозрядних чисел дорівнює $3(3N\log_2 N + 2N + 3C_{FFT})$.

Теорема доведена.

Приклад з обчисленням ДПФ у полі комплексних чисел. Множення дворозрядних чисел $12\cdot 31 = 372$ представимо у вигляді векторів-стовпців: $X_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$, $Y_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$. Всі операції відбуваються з цілими числами, тому операція заокруглення (крок 5) не позначена.

$$W_{4,4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix},\ X_4 = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \end{bmatrix},\ Y_4 = \begin{bmatrix} 1 \\ 3 \\ 0 \\ 0 \end{bmatrix},\ \hat{X}_4 = W_{4,4}\cdot X_4 = \begin{bmatrix} 3 \\ 2-i \\ 1 \\ 2+i \end{bmatrix},\ \hat{Y}_4 = W_{4,4}\cdot Y_4 = \begin{bmatrix} 4 \\ 1-3i \\ -2 \\ 1+3i \end{bmatrix},$$

$$\hat{Z}_4 = \hat{X}_4\cdot\hat{Y}_4 = \begin{bmatrix} 12 \\ -1-7i \\ -2 \\ -1+7i \end{bmatrix},\ Z_4 = \frac{1}{4}\cdot W_{4,4}\cdot\hat{Z}_4^* = \frac{1}{4}\cdot W_{4,4}\cdot\begin{bmatrix} 12 \\ -1+7i \\ -2 \\ -1-7i \end{bmatrix} = \frac{1}{4}\cdot\begin{bmatrix} 8 \\ 28 \\ 12 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 3 \\ 0 \end{bmatrix}.$$

Взаємозв'язок між дискретними перетвореннями. Дискретні косинусні, синусні перетворення та дискретне перетворення Фур'є взаємозв'язані між собою. Виразимо ДКП та ДСП через ДПФ.

Формула Ейлера стверджує, що для будь-якого дійсного числа x виконується рівність $e^{ix} = \cos x + i\sin x$. За допомогою формули Ейлера можна представити функції $\cos x$ та $\sin x$ у наступному вигляді:

$$\cos x = \mathrm{Re}(e^{ix}) = \frac{e^{ix} + e^{-ix}}{2},\ \sin x = \mathrm{Im}(e^{ix}) = \frac{e^{ix} - e^{-ix}}{2i}. \tag{2}$$

Використаємо наступні позначення: $C_{2N}^{\langle k\cdot r\rangle_{2N}} = \cos(\frac{k\cdot r\cdot\pi}{2N})$, $S_{2N}^{\langle k\cdot r\rangle_{2N}} = \sin(\frac{k\cdot r\cdot\pi}{2N})$.

З урахуванням (2) можна записати, що

$$C_{2N}^{\langle k\cdot r\rangle_{2N}} = \cos(\frac{k\cdot r\cdot\pi}{2N}) = \frac{1}{2}(e^{\frac{k\cdot r\cdot\pi}{2N}i} + e^{-\frac{k\cdot r\cdot\pi}{2N}i}) = \frac{1}{2}(W_{2N}^{-\langle k\cdot r\rangle_{2N}} + W_{2N}^{\langle k\cdot r\rangle_{2N}}) = \frac{1}{2}((W_{2N}^{\langle k\cdot r\rangle_{2N}})^* + W_{2N}^{\langle k\cdot r\rangle_{2N}}),$$

$$S_{2N}^{\langle k\cdot r\rangle_{2N}} = \sin(\frac{k\cdot r\cdot\pi}{2N}) = \frac{1}{2i}(e^{\frac{k\cdot r\cdot\pi}{2N}i} - e^{-\frac{k\cdot r\cdot\pi}{2N}i}) = \frac{1}{2i}(W_{2N}^{-\langle k\cdot r\rangle_{2N}} - W_{2N}^{\langle k\cdot r\rangle_{2N}}) = \frac{1}{2i}((W_{2N}^{\langle k\cdot r\rangle_{2N}})^* - W_{2N}^{\langle k\cdot r\rangle_{2N}}).$$

За аналогією з матрицею ДПФ $W_{2N,2N}$ розмірами $2N$ на $2N$ можна записати матриці $C_{2N,2N}$ та $S_{2N,2N}$, які мають наступні елементи:

$$C_{2N}^{\langle k\cdot r\rangle_{2N}} = \cos(\frac{k\cdot r\cdot\pi}{2N}),\ S_{2N}^{\langle k\cdot r\rangle_{2N}} = \sin(\frac{k\cdot r\cdot\pi}{2N}),\ k,r = \overline{0, 2N-1}.$$

Тоді додавання та різницю матриць ДПФ можна записати у наступному вигляді:

$$\frac{1}{2}(W^*_{2N,2N} - W_{2N.2N}) = C_{2N.2N}\,,\ \frac{1}{2i}(W^*_{2N,2N} + W_{2N.2N}) = S_{2N.2N}\,. \tag{3}$$

З виразу (3) можна обчислити $W_{2N,2N}$ та $W^*_{2N,2N}$ наступним чином:

$$W_{2N.2N} = C_{2N.2N} - iS_{2N.2N}\,,\ W^*_{2N,2N} = C_{2N.2N} + iS_{2N.2N}\,.$$

Використання властивостей ДПФ дійсних сигналів. Використання наступних властивостей ДПФ дійсного сигналу дозволяє значно зменшити кількість комплексних множень

$$\operatorname{Im}\hat{Z}_{2N}(0) = 0\,,\ \operatorname{Im}\hat{Z}_{2N}(N) = 0\,;$$

$$\operatorname{Re}\hat{Z}_{2N}(r) = \operatorname{Re}\hat{Z}_{2N}(2N-r)\,,\ \operatorname{Im}\hat{Z}_{2N}(r) = -\operatorname{Im}\hat{Z}_{2N}(2N-r)\,,\ r = \overline{1, N-1}\,. \tag{4}$$

Лема 1. Для будь-якого комплексного сигналу $\hat{Z}_{2N}$ розрядністю $2N$, який має властивість (4), правильний наступний вираз: $W^*_{2N,2N} \cdot \hat{Z}_{2N} = C_{2N,2N} \cdot \operatorname{Re}\hat{Z}_{2N} - S_{2N,2N} \cdot \operatorname{Im}\hat{Z}_{2N}$.

Доведення. Обчислення ОДПФ від сигналу $\hat{Z}_{2N}$ можна записати наступним чином:

$$Z_{2N}(r) = \frac{1}{2N} \cdot W^*_{2N,2N} \cdot \hat{Z}_{2N} = \frac{1}{2N} \cdot \sum_{k=0}^{2N-1} ((W_{2N}^{\langle k\cdot r\rangle_{2N}})^* \cdot \hat{Z}_{2N}(k)) = \frac{1}{2N} \cdot \sum_{k=0}^{2N-1} ((C_{2N}^{\langle k\cdot r\rangle_{2N}} + iS_{2N}^{\langle k\cdot r\rangle_{2N}}) \cdot \hat{Z}_{2N}(k)),$$

$$r = \overline{0, 2N-1}\,.$$

З урахуванням (4) попередній вираз можна представити у наступному вигляді:

$$Z_{2N}(r) = \frac{1}{2N} \cdot C_{2N}^{\langle 0\cdot r\rangle_{2N}} \cdot \hat{Z}_{2N}(0) + \frac{1}{2N} \cdot C_{2N}^{\langle N\cdot r\rangle_{2N}} \cdot \hat{Z}_{2N}(N) +$$

$$+\frac{1}{2N} \cdot \sum_{k=1}^{N-1} ((C_{2N}^{\langle k\cdot r\rangle_{2N}} + iS_{2N}^{\langle k\cdot r\rangle_{2N}}) \cdot \hat{Z}_{2N}(k)) + \frac{1}{2N} \cdot \sum_{k=1}^{N-1} ((C_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}} + iS_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}}) \cdot \hat{Z}_{2N}(2N-k)),$$

$$r = \overline{0, 2N-1}\,.$$

Візьмемо до уваги, що $C_{2N}^{\langle k\cdot r\rangle_{2N}} = C_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}}$, $S_{2N}^{\langle k\cdot r\rangle_{2N}} = -S_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}}$, виходячи з властивостей тригонометричних функцій

$$C_{2N}^{\langle k\cdot r\rangle_{2N}} = \cos(\frac{k\cdot r\cdot \pi}{2N}) = \cos(\frac{-k\cdot r\cdot \pi}{2N}) = \cos(\frac{(2N-k)\cdot r\cdot \pi}{2N}) = C_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}}\,,$$

$$S_{2N}^{\langle k\cdot r\rangle_{2N}} = \sin(\frac{k\cdot r\cdot \pi}{2N}) = -\sin(\frac{-k\cdot r\cdot \pi}{2N}) = -\sin(\frac{(2N-k)\cdot r\cdot \pi}{2N}) = -S_{2N}^{\langle (2N-k)\cdot r\rangle_{2N}}\,. \tag{5}$$

З урахуванням (5) можна записати наступне:

$$Z_{2N}(r) = \frac{1}{2N} \cdot C_{2N}^{\langle 0\cdot r\rangle_{2N}} \cdot \hat{Z}_{2N}(0) + \frac{1}{2N} \cdot C_{2N}^{\langle N\cdot r\rangle_{2N}} \cdot \hat{Z}_{2N}(N) +$$

$$+\frac{1}{2N} \cdot \sum_{k=1}^{N-1} ((C_{2N}^{\langle k\cdot r\rangle_{2N}} + iS_{2N}^{\langle k\cdot r\rangle_{2N}}) \cdot \hat{Z}_{2N}(k)) + \frac{1}{2N} \cdot \sum_{k=1}^{N-1} ((C_{2N}^{\langle (k)\cdot r\rangle_{2N}} - iS_{2N}^{\langle (k\cdot r\rangle_{2N}}) \cdot \hat{Z}^*_{2N}(k))\,,\ r = \overline{0, 2N-1}\,.$$

Після групування отримуємо

$$Z_{2N}(r)=\frac{1}{2N}\cdot C_{2N}^{\langle 0\cdot r\rangle_{2N}}\cdot\hat{Z}_{2N}(0)+\frac{1}{2N}\cdot C_{2N}^{\langle N\cdot r\rangle_{2N}}\cdot\hat{Z}_{2N}(N)+$$

$$+\frac{1}{2N}\cdot\sum_{k=1}^{N-1}(C_{2N}^{\langle k\cdot r\rangle_{2N}}\cdot(\hat{Z}_{2N}(k)+\hat{Z}_{2N}^{*}(k))+\frac{1}{2N}\cdot iS_{2N}^{\langle k\cdot r\rangle_{2N}}\cdot(\hat{Z}_{2N}(k)-\hat{Z}_{2N}^{*}(k)))\,,\ r=\overline{0,2N-1}.$$

$$Z_{2N}(r)=\frac{1}{2N}\cdot C_{2N}^{\langle 0\cdot r\rangle_{2N}}\cdot\hat{Z}_{2N}(0)+\frac{1}{2N}\cdot C_{2N}^{\langle N\cdot r\rangle_{2N}}\cdot\hat{Z}_{2N}(N)+$$

$$+\frac{1}{2N}\cdot 2\sum_{k=1}^{N-1}(C_{2N}^{\langle k\cdot r\rangle_{2N}}\cdot\operatorname{Re}\hat{Z}_{2N}(k)+iS_{2N}^{\langle k\cdot r\rangle_{2N}}\cdot i\operatorname{Im}\hat{Z}_{2N}(k))\,,\ r=\overline{0,2N-1}.$$

Остаточно у загальному вигляді отримуємо

$$Z_{2N}=\frac{1}{2N}\cdot W_{2N,2N}^{*}\cdot\hat{Z}_{2N}=\frac{1}{2N}\cdot C_{2N,2N}\cdot\operatorname{Re}\hat{Z}_{2N}-\frac{1}{2N}\cdot S_{2N,2N}\cdot\operatorname{Im}\hat{Z}_{2N}\,.$$

Опускаючи множник $1/2N$, отримуємо необхідний вираз.

Лема доведена.

Наступною властивістю є те, що результат поелементного множення двох ДПФ дійсних сигналів також має властивість (4):

$$\hat{Z}_{2N}(0)\leftarrow\operatorname{Re}\hat{X}_{2N}(0)\cdot\operatorname{Re}\hat{Y}_{2N}(0)\,,$$

$$\hat{Z}_{2N}(r)\leftarrow\hat{X}_{2N}(r)\cdot\hat{Y}_{2N}(r)\,,\ r=\overline{1,N-1}\,,$$

$$\hat{Z}_{2N}(N)\leftarrow\operatorname{Re}\hat{X}_{2N}(N)\cdot\operatorname{Re}\hat{Y}_{2N}(N)\,,$$

$$\hat{Z}_{2N}(2N-r)\leftarrow\hat{Z}_{2N}^{*}(r)\,,\ r=\overline{1,N-1}\,. \tag{6}$$

Алгоритм множення з обчисленням ДПФ у полі дійсних чисел. При заміні матриці $W_{2N,2N}$ у Алгоритмі 1 матрицями $C_{2N,2N}$ та $S_{2N,2N}$ можна побудувати наступний алгоритм множення багаторозрядних чисел.

Алгоритм 2. Множення двох N-розрядних чисел з обчисленням ДПФ розрядністю $2N$ у полі дійсних чисел з використанням матриць $C_{2N,2N}$ та $S_{2N,2N}$.

1. Додавання старших нулів: $X_{2N}(N+r)\leftarrow Y_{2N}(N+r)\leftarrow 0$, $r=\overline{0,N-1}$.
2. Обчислення **ДПФ розрядністю** $2N$:

$$\hat{X}_{2N}\leftarrow(C_{2N.2N}-iS_{2N.2N})\cdot X_{2N}\,,\ \hat{Y}_{2N}\leftarrow(C_{2N.2N}-iS_{2N.2N})\cdot Y_{2N}\,.$$

3. Перемноження **ДПФ розрядністю** $2N$: $\hat{Z}_{2N}(r)\leftarrow\hat{X}_{2N}(r)\cdot\hat{Y}_{2N}(r)$, $r=\overline{0,2N-1}$.
4. Обчислення **ОДПФ**: $Z_{2N}\leftarrow\frac{1}{2N}\,(C_{2N.2N}\cdot\operatorname{Re}\hat{Z}_{2N}-S_{2N.2N}\cdot\operatorname{Im}\hat{Z}_{2N})$.
5. Обчислення результату: $Z_{2N}(r)\leftarrow[Z_{2N}(r)]$, $r=\overline{0,2N-1}$.

Похибка заокруглення в Алгоритмі 2 така ж, як і в Алгоритмі 1.

Перевагою Алгоритму 2 є те, що обчислення дійсних ($C_{2N,2N} \cdot X_{2N}$, $C_{2N,2N} \cdot Y_{2N}$) та комплексних ($S_{2N,2N} \cdot X_{2N}$, $S_{2N,2N} \cdot Y_{2N}$) частин ДПФ та ОДПФ від $\hat{Z}_{2N}(r)$ ($C_{2N.2N} \cdot \mathrm{Re}\hat{Z}_{2N}$, $S_{2N.2N} \cdot \mathrm{Im}\hat{Z}_{2N}$) на кроках 2 та 4 може бути розділено. Тобто обчислення відбуваються у полі дійсних чисел замість обчислення у полі комплексних чисел.

Теорема 2. Складність Алгоритму 2 за кількістю операцій множення однорозрядних дійсних чисел має наступний вигляд:

$$O_{\mathrm{Algorithm\ 2}}^{multiplication}(2N) = 6N\log_2 N + 6N + 6C_{ДКП-ДСП}.$$

Доведення. На кроці 2 для обчислення ДПФ необхідно матриці $C_{2N,2N}$ та $S_{2N,2N}$ розмірами $2N$ на $2N$ дійсних чисел помножити на вектор-стовбець довжиною $2N$ дійсних чисел. Складність множення матриці $C_{2N,2N}$ на вектор-стовбець X_{2N} за кількістю операцій множення однорозрядних дійсних чисел є $(2N)^2$. За аналогією з ДКП обчислення $C_{2N,2N}$ та $S_{2N,2N}$ може бути пришвидшено до $N\log_2 N + C_{ДКП-ДСП}$, де $C_{ДКП-ДСП}$ – константа. Тоді на кроці 2 обчислення двох ДПФ потребує $4N\log_2 2N + 4C_{ДКП-ДСП}$ операцій множення дійсних однорозрядних чисел при множенні матриць $C_{2N,2N}$ та $S_{2N,2N}$ на вектор-стовпці X_{2N} та Y_{2N}.

На кроці 3 поелементне множення двох вектор-стовпців довжиною $2N$ комплексних чисел потребує $2N$ операцій множення комплексних чисел. Аналогічно алгоритму 1 достатньо $3 \cdot 2N$ операцій множення дійсних чисел для обчислення кроку 3.

На кроці 4 для обчислення ОДПФ необхідно матриці $C_{2N,2N}$ та $S_{2N,2N}$ помножити на вектор-стовпці, що є дійсними та уявними частинами $\hat{Z}_{2N}$, і це потребує $2N\log_2 N + 2C_{ДКП-ДСП}$ множень однорозрядних дійсних чисел за аналогією з кроком 2.

Загальна кількість операцій множення дійсних чисел відповідає виразу

$$4N\log_2 N + 4C_{ДКП-ДСП} + 6N + 2N\log_2 N + 2C_{ДКП-ДСП} = 6N\log_2 N + 6N + 6C_{ДКП-ДСП}.$$

Теорема доведена.

Основна ідея Алгоритму 2 це заміна матриці $W_{2N,2N}$ матрицями $C_{2N,2N}$ та $S_{2N,2N}$, що дозволяє розділити обчислення дійсної та уявної частин ДПФ.

Розглядаючи загальну кількість однорозрядних операцій множення, Алгоритм 2 виконує приблизно 67 % однорозрядних операцій множення у порівнянні з алгоритмом 1, якщо вважати N доволі великим числом та не враховувати доданки $6N$ та константи $9C_{FFT}$, $6C_{ДКП-ДСП}$ отримуємо, що

$$(6N\log_2 N + 6N + 6C_{ДКП-ДСП})/(9N\log_2 N + 6N + 9C_{FFT}) \approx 6/9 \approx 0{,}67.$$

Приклад з обчисленням ДПФ у полі дійсних чисел. Множення чисел $1112 \cdot 1121 = 1246552$ можна представити у вигляді

$$X_4 = \begin{bmatrix} 2 & 1 & 1 & 1 \end{bmatrix}^T,\ Y_4 = \begin{bmatrix} 1 & 2 & 1 & 1 \end{bmatrix}^T,\ q = 1/\sqrt{2}.$$

Тоді $X_8 = \begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T U$, $Y_8 = \begin{bmatrix} 1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T$

$$C_{8,8} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & q & 0 & -q & -1 & -q & 0 & q \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ 1 & -q & 0 & q & -1 & q & 0 & -q \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -q & 0 & q & -1 & q & 0 & -q \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ 1 & q & 0 & -q & -1 & -q & 0 & q \end{bmatrix}, S_{8,8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q & 1 & q & 0 & -q & -1 & -q \\ 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 \\ 0 & q & -1 & q & 0 & -q & 1 & -q \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -q & 1 & -q & 0 & q & -1 & q \\ 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 \\ 0 & -q & -1 & -q & 0 & q & 1 & q \end{bmatrix}.$$

$$\hat{X}_8 = \begin{bmatrix} 5 \\ 2-2.4142i \\ 1 \\ 2-0.4142i \\ 1 \\ 2+0.4142i \\ 1 \\ 2+2.4142i \end{bmatrix},\ \hat{Y}_8 = \begin{bmatrix} 5 \\ 1.7071-3.1213i \\ -i \\ 0.2929-1.1213i \\ -1 \\ 0.2929+1.1213i \\ i \\ 1.7071+3.1213i \end{bmatrix},\ \hat{Z}_8 = \hat{X}_8 \cdot \hat{Y}_8 = \begin{bmatrix} 25 \\ -4.12124-10.36388i \\ -i \\ 0.121358-2.363919i \\ -1 \\ 0.121358+2.36392i \\ i \\ -4.12124+10.3639i \end{bmatrix},$$

$$Z_8 = [\frac{1}{8} \cdot (C_{8.8} \cdot \operatorname{Re}\hat{Z}_8 - S_{8.8} \cdot \operatorname{Im}\hat{Z}_8)] = [\frac{1}{8} \cdot \begin{bmatrix} 16.0002-0 \\ 20.0001-(-19.9997) \\ 24-(-15.9999) \\ 31.9999-(-15.9997) \\ 31.9998-0 \\ 31.9998-15.9997 \\ 24-15.9999 \\ 20.0001-19.9997 \end{bmatrix}] = \begin{bmatrix} [2.0000] \\ [5.0000] \\ [5.0000] \\ [5.9999] \\ [4.0000] \\ [2.0000] \\ [1.0000] \\ [0.0001] \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 5 \\ 6 \\ 4 \\ 2 \\ 1 \\ 0 \end{bmatrix}.$$

Алгоритм множення на основі ДКП та ДСП. У Алгоритмі 2 вектор-стовпці $\hat{X}_{2N}$, $\hat{Y}_{2N}$, $\hat{Z}_{2N}$ використовуються тільки для зберігання комплексних елементів. На всіх кроках алгоритму можливе виключення комплексних операцій за рахунок введення вектор-стовпців $\hat{X}R_{2N}$, $\hat{Y}R_{2N}$, $\hat{Z}R_{2N}$ та $\hat{X}I_{2N}$, $\hat{Y}I_{2N}$, $\hat{Z}I_{2N}$, які б зберігали окремо дійсну та уявну частину вектор-стовпців $\hat{X}_{2N}$, $\hat{Y}_{2N}$, $\hat{Z}_{2N}$. Для пришвидшення врахуємо, що вектор-стовпці $\hat{X}_{2N}$, $\hat{Y}_{2N}$, $\hat{Z}_{2N}$ мають симетрію $\begin{bmatrix} a & b & c & d & e & d^* & c^* & b^* \end{bmatrix}^T$ для $N = 4$, а матриці $C_{2N,2N}$ та $S_{2N,2N}$ за виключенням нульових рядків та стовпців мають симетрію відносно N-го рядку та N-го стовпця. Тому достатньо множити на матриці $C_{N+1,N+1}$ та $S_{N+1,N+1}$, елементи яких отримані з $N+1$ перших рядків

та стовпців матриць $C_{2N,2N}$ та $S_{2N,2N}$. Використання властивості (6) дає можливість зменшити майже у два рази кількість обчислень на кроках 2, 3 та 4 у наступному алгоритмі.

Алгоритм 3. Множення двох N -розрядних чисел на основі ДКП та ДСП.

1. Додавання старших нулів: $X_{N+1}(N) \leftarrow Y_{N+1}(N) \leftarrow 0$.
2. Обчислення **ДКП, ДСП**:

$$\widehat{X}R_{N+1} \leftarrow C_{N+1.N+1} \cdot X_{N+1},\ \widehat{X}I_{N+1} \leftarrow S_{N+1.N+1} \cdot X_{N+1},$$

$$\widehat{Y}R_{N+1} \leftarrow C_{N+1.N+1} \cdot Y_{N+1},\ \widehat{Y}I_{N+1} \leftarrow S_{N+1.N+1} \cdot Y_{N+1}.$$

$$\widehat{X}_{N+1} \leftarrow \widehat{X}R_{N+1} - i \cdot \widehat{X}I_{N+1},\ \widehat{Y}_{N+1} \leftarrow \widehat{Y}R_{N+1} - i \cdot \widehat{Y}I_{N+1}.$$

3. Поелементне множення:

$$\widehat{Z}_{N+1}(r) \leftarrow \widehat{X}_{N+1}(r) \cdot \widehat{Y}_{N+1}(r),$$

$$\widehat{Z}R_{N+1}(r) \leftarrow \operatorname{Re}\widehat{Z}_{N+1}(r),\ \widehat{Z}R_{N+1}(r) \leftarrow \operatorname{Im}\widehat{Z}_{N+1}(r),\ r = \overline{0, N}.$$

4. Обчислення **ОДКП**, **ОДСП**:

$$ZR_{N+1}(r) \leftarrow \sum_{k=0}^{N} (d(k) \cdot C_{N+1.N+1}(r,k) \cdot \widehat{Z}R_{N+1}(r)),$$

$$ZI_{N+1}(r) \leftarrow \sum_{k=0}^{N} (d(k) \cdot S_{N+1.N+1}(r,k) \cdot \widehat{Z}I_{N+1}(r)),$$

$$d(k) = \begin{cases} 1 & \text{якщо } k = 0 \\ (-1)^r & \text{якщо } k = N \\ 2 & \text{якщо } k \neq 0 \text{ або } k \neq N \end{cases},\ r = \overline{0, N}.$$

5. Обчислення результату **розрядністю** $2N$:

$$ZR_{2N}(r) \leftarrow ZR_{N+1}(r),\ ZI_{2N}(r) \leftarrow ZI_{N+1}(r),\ r = \overline{0, N}.$$

$$ZR_{2N}(2N - r) \leftarrow ZR_{2N}(r),\ ZI_{2N}(2N - r) \leftarrow -ZI_{2N}(r),\ r = \overline{1, N-1}.$$

$$Z_{2N} \leftarrow \frac{1}{2N}\ (ZR_{2N} - ZI_{2N}).$$

$$Z_{2N}(r) \leftarrow [Z_{2N}(r)],\ r = \overline{0, 2N-1}.$$

В Алгоритмі 3 на кроці 2 обчислення відповідають Discrete Cosine Transform (DCT) та Discrete Sine Transform (DST) першого типу

$$U_r \leftarrow l \cdot (u_0 + (-1)^r u_{N_{DCT}}) + \sum_{k=1}^{N_{DCT}-1} u_k \cos(\frac{\pi \cdot k \cdot r}{N_{DCT}}),\ r = \overline{0, N_{DCT}},$$

де N_{DCT} – парне, $l=1$ – відповідає ДПФ дійсного сигналу розрядності $2N_{DCT}$, у якого старші розряди нульові.

$$V_r \leftarrow \sum_{k=0}^{N_{DST}-2} v_k \sin(\frac{\pi\cdot(k+1)\cdot(r+1)}{N_{DST}}),\ r=\overline{0, N_{DST}-2},\ N_{DST} \text{ – парне.}$$

На кроці 4 обчислення відповідають Inverse Discrete Cosine Transform (IDCT) та Inverse Discrete Sine Transform (IDST) першого типу.

$$u_r \leftarrow \frac{1}{N}(\frac{1}{2}(U_0+(-1)^r U_{N_{IDCT}})+\sum_{k=1}^{N_{IDCT}-1} U_k \cos(\frac{\pi\cdot k\cdot r}{N_{IDCT}})),\ r=\overline{0, N_{IDCT}},\ N_{IDCT} \text{ – парне.}$$

$$v_r \leftarrow \frac{1}{N}\sum_{k=0}^{N_{DST}-2} V_k \sin(\frac{\pi\cdot(k+1)\cdot(r+1)}{N_{IDST}}),\ r=\overline{0, N_{IDST}-2},\ N_{IDST} \text{ – парне.}$$

Матриця $S_{N+1,N+1}$ має додаткові нульові рядки та стовпці для зручності обчислення.

Приклад обчислення на основі ДКП та ДСП. Множення чисел $1112\cdot 1121=1246552$ можна представити у вигляді $X_4=\begin{bmatrix}2 & 1 & 1 & 1\end{bmatrix}^T$, $Y_4=\begin{bmatrix}1 & 2 & 1 & 1\end{bmatrix}^T$, $q=1/\sqrt{2}$.

Тоді $X_5=\begin{bmatrix}2 & 1 & 1 & 1 & 0\end{bmatrix}^T$, $Y_5=\begin{bmatrix}1 & 2 & 1 & 1 & 0\end{bmatrix}^T$.

$$C_{5,5}=\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & q & 0 & -q & -1\\ 1 & 0 & -1 & 0 & 1\\ 1 & -q & 0 & q & -1\\ 1 & -1 & 1 & -1 & 1\end{bmatrix},\ S_{5,5}=\begin{bmatrix}0 & 0 & 0 & 0 & 0\\ 0 & q & 1 & q & 0\\ 0 & 1 & 0 & -1 & 0\\ 0 & q & -1 & q & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}.$$

$$\hat{X}R_5=\begin{bmatrix}5\\2\\1\\2\\1\end{bmatrix},\hat{X}I_5=\begin{bmatrix}0\\-2.4142\\0\\-0.4142\\0\end{bmatrix},\hat{Y}R_5=\begin{bmatrix}5\\1.7071\\0\\0.2929\\-1\end{bmatrix},\hat{Y}I_5=\begin{bmatrix}0\\-3.1213\\1\\-1.1213\\0\end{bmatrix},\hat{Z}R_5=\begin{bmatrix}25\\-4.12124\\0\\0.121358\\-1\end{bmatrix},\hat{Z}I_5=\begin{bmatrix}0\\-10.36388\\-1\\-2.363919\\0\end{bmatrix}.$$

$$ZR_5=\begin{bmatrix}16.0002 & 20.0001 & 24.0000 & 31.9999 & 31.9998\end{bmatrix}^T,$$

$$ZI_5=\begin{bmatrix}0.0000 & -19.9997 & -15.9999 & -15.9997 & 0.0000\end{bmatrix}^T,$$

$$ZR_8(r)\leftarrow ZR_5(r),\ ZI_8(r)\leftarrow ZI_5(r),\ r=\overline{0,4}.$$

$$ZR_8(2N-r)\leftarrow ZR_8(r),\ ZI_8(2N-r)\leftarrow -ZI_8(r),\ r=\overline{1,3}.$$

$$Z_8 = [\frac{1}{2\cdot 4}\cdot(ZR_8 - ZI_8)] = [\frac{1}{2\cdot 4}\cdot\begin{bmatrix} 16.0002-0 \\ 20.0001-(-19.9997) \\ 24-(-15.9999) \\ 31.9999-(-15.9997_ \\ 31.9998-0 \\ 31.9998-15.9997 \\ 24-15.9999 \\ 20.0001-19.9997 \end{bmatrix}] = \begin{bmatrix} [2.0000] \\ [5.0000] \\ [5.0000] \\ [5.9999] \\ [4.0000] \\ [2.0000] \\ [1.0000] \\ [0.0001] \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 5 \\ 6 \\ 4 \\ 2 \\ 1 \\ 0 \end{bmatrix}.$$

Алгоритм множення на основі ДКП та ДСП у полі цілих чисел. В Алгоритмі 3 обчислення $\hat{X}R_{2N}$, $\hat{Y}R_{2N}$, $\hat{Z}R_{2N}$ та $\hat{X}I_{2N}$, $\hat{Y}I_{2N}$, $\hat{Z}I_{2N}$ можуть бути переведені у поле цілих чисел при використанні коефіцієнта $M = 2^m$, де m – ціле число, при розрахунку елементів матриць $C_{N+1,N+1}$ та $S_{N+1,N+1}$ з подальшим їх заокругленням

$$C_{N+1}^{\langle k\cdot r\rangle_{2N}} = [M\cdot\cos(\frac{\pi\cdot k\cdot r}{2N})],\ S_{N+1}^{\langle k\cdot r\rangle_{2N}} = [M\cdot\sin(\frac{\pi\cdot k\cdot r}{2N})],\ k,r=\overline{0,N}. \quad (7)$$

За аналогією з методом стиснення зображень за допомогою ДКП множення на матриці $[M\cdot C_{N+1,N+1}]$ та $[M\cdot S_{N+1,N+1}]$ (елементи яких розраховані на основі (7)) може бути реалізоване за допомогою бітових зсувів, додавань та віднімань без використання операцій однорозрядного множення. Далі обчислення ДКП, ДСП, ОДКП та ОДСП позначені, як результат операцій множення, але вважаємо, що їх обчислення відбувається на основі бітових зсувів, додавань та віднімань.

Введемо додаткові коефіцієнти $K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$, $K_2 = 2^{k_2}$, де k_0, k_1, k_2 – цілі числа, та використаємо їх у наступному алгоритмі.

Алгоритм 4. Множення двох N-розрядних чисел на основі ДКП та ДСП у полі цілих чисел.

1. Ініціалізація:

$$X_{N+1}(N) \leftarrow Y_{N+1}(N) \leftarrow 0.$$

$$RC_{N+1.N+1} \leftarrow [K_0\cdot K_1\cdot C_{N+1.N+1}],\ RS_{N+1.N+1} \leftarrow [K_0\cdot K_1\cdot S_{N+1.N+1}].$$

2. Обчислення **ДКП**, **ДСП**, де $K_3 = K_2/K_0$:

$$\hat{X}R_{N+1} \leftarrow [\frac{1}{K_3}\cdot RC_{N+1.N+1}\cdot X_{N+1}],\ \hat{X}I_{N+1} \leftarrow [\frac{-1}{K_3}\cdot RS_{N+1.N+1}\cdot X_{N+1}],$$

$$\hat{Y}R_{N+1} \leftarrow [\frac{1}{K_3}\cdot RC_{N+1.N+1}\cdot Y_{N+1}],\ \hat{Y}I_{N+1} \leftarrow [\frac{-1}{K_3}\cdot RS_{N+1.N+1}\cdot Y_{N+1}].$$

3. Поелементне множення:

$$\hat{Z}R_{N+1}(r) \leftarrow [\frac{1}{K_1\cdot K_1}\cdot(\hat{X}R_{N+1}(r)\cdot\hat{Y}R_{N+1}(r) - \hat{X}I_{N+1}(r)\cdot\hat{Y}I_{N+1}(r))],$$

$$\hat{Z}I_{N+1}(r) \leftarrow [\frac{1}{K_1\cdot K_1}\cdot(\hat{X}R_{N+1}(r)\cdot\hat{Y}I_{N+1}(r) + \hat{X}I_{N+1}(r)\cdot\hat{Y}R_{N+1}(r))],\ r=\overline{0,N}.$$

4. Обчислення **ОДКП**, **ОДСП**:

$$ZR_{N+1}(r) \leftarrow \sum_{k=0}^{N} (d(k) \cdot RC_{N+1.N+1}(r,k) \cdot \widehat{Z}R_{N+1}(r)),$$

$$ZI_{N+1}(r) \leftarrow \sum_{k=0}^{N} (d(k) \cdot RS_{N+1.N+1}(r,k) \cdot \widehat{Z}I_{N+1}(r)),$$

$$d(k) = \begin{cases} 1 & \text{якщо } k = 0 \\ (-1)^r & \text{якщо } k = N \\ 2 & \text{якщо } k \neq 0 \text{ або } k \neq N \end{cases}, \quad r = \overline{0, N}.$$

5. Обчислення результату **розрядністю** $2N$, де $K_4 = 2 \cdot N \cdot (K_0 \cdot K_1) \cdot K_2 \cdot K_2$:

$$ZR_{2N}(r) \leftarrow ZR_{N+1}(r)],\ ZI_{2N}(r) \leftarrow ZI_{N+1}(r),\ r = \overline{0, N}.$$

$$ZR_{2N}(2N - r) \leftarrow ZR_{2N}(r),\ ZI_{2N}(2N - r) \leftarrow -ZI_{2N}(r),\ r = \overline{1, N-1}.$$

$$Z_{2N} \leftarrow [\frac{1}{K_4} \cdot (ZR_{2N} - ZI_{2N})].$$

Приклад обчислення на основі ДКП та ДСП у полі цілих чисел. Множення чисел $1112 \cdot 1121 = 1246552$ можна представити у вигляді $X_4 = [2 \quad 1 \quad 1 \quad 1]^T$, $Y_4 = [1 \quad 2 \quad 1 \quad 1]^T$, $K_0 = K_2 = 1$, $K_1 = 256$.

Тоді $X_5 = [2 \quad 1 \quad 1 \quad 1 \quad 0]^T$, $Y_5 = [1 \quad 2 \quad 1 \quad 1 \quad 0]^T$.

$$RC_{5,5} = \begin{bmatrix} 256 & 256 & 256 & 256 & 256 \\ 256 & 181 & 0 & -181 & -256 \\ 256 & 0 & -256 & 0 & 256 \\ 256 & -181 & 0 & 181 & -256 \\ 256 & -256 & 256 & -256 & 256 \end{bmatrix},\ RS_{5,5} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 181 & 256 & 181 & 0 \\ 0 & 256 & 0 & -256 & 0 \\ 0 & 181 & -256 & 181 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

$$\widehat{X}R_5 = \begin{bmatrix} 1280 \\ 512 \\ 256 \\ 512 \\ 256 \end{bmatrix},\ \widehat{X}I_5 = \begin{bmatrix} 0 \\ -618 \\ 0 \\ -106 \\ 0 \end{bmatrix},\ \widehat{Y}R_5 = \begin{bmatrix} 1280 \\ 437 \\ 0 \\ 75 \\ -256 \end{bmatrix},\ \widehat{Y}I_5 = \begin{bmatrix} 0 \\ -799 \\ -256 \\ -287 \\ 0 \end{bmatrix}.$$

$$\widehat{Z}R_5 = [\frac{1}{256 \cdot 256} \cdot \begin{bmatrix} 1638400 \\ -270038 \\ 0 \\ 7978 \\ -65536 \end{bmatrix}] = \begin{bmatrix} 25 \\ -4 \\ 0 \\ 0 \\ -1 \end{bmatrix},\ \widehat{Z}I_5 = [\frac{1}{256 \cdot 256} \cdot \begin{bmatrix} 0 \\ -679054 \\ -65536 \\ -154894 \\ 0 \end{bmatrix}] = \begin{bmatrix} 0 \\ -10 \\ -1 \\ -2 \\ 0 \end{bmatrix}.$$

$$ZR_5 = [4096 \quad 5208 \quad 6144 \quad 8104 \quad 8192]^T,$$

$$ZI_5 = [0 \quad -4856 \quad -4096 \quad -3832 \quad 0]^T.$$

$$ZR_8(r) \leftarrow ZR_5(r),\ ZI_8(r) \leftarrow ZI_5(r),\ r = \overline{0,4}.$$

$$ZR_8(2N-r) \leftarrow ZR_8(r),\ ZI_8(2N-r) \leftarrow -ZI_8(r),\ r = \overline{1,3}.$$

$$Z_8 = [\frac{1}{2\cdot 4\cdot 256}\cdot(ZR_8 - ZI_8)] = [\frac{1}{2\cdot 4\cdot 256}\cdot\begin{bmatrix} 4096-0 \\ 5208-(-4856) \\ 6144-(-4096) \\ 8104-(-3832) \\ 8192-0 \\ 8104-3832 \\ 6144-4096 \\ 5208-4856 \end{bmatrix}] = \begin{bmatrix} [2] \\ [4.9141] \\ [5] \\ [5.8281] \\ [4] \\ [2.0859] \\ [1] \\ [0.1719] \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 5 \\ 6 \\ 4 \\ 2 \\ 1 \\ 0 \end{bmatrix}.$$

В Алгоритмі 4 множення на коефіцієнт $1/2N$, яке необхідне для обчислення оберненого перетворення, перенесено з кроку 4 до кроку 5 для того, щоб мінімізувати виконання операції ділення, яка реалізується за рахунок операцій бітового зсуву.

Якщо в алгоритмі 4 не враховувати заокруглення та прийняти, що $K_0 = K_1 = K_2 = 1$, то обчислення алгоритму 4 не відрізняється від обчислення Алгоритму 3.

Алгоритм на перших кроках оперує невеликими числами за довжиною (у бітах), але на кожному кроці довжина у бітах збільшується, що призводить до того, що довжини слова у 32, 64, 128 і т. д. бітів не вистачає.

Теорема 3. Алгоритм 4 множення двох N-розрядних чисел на основі ДКП та ДСП в послідовній моделі обчислень потребує $3N-1$ операцій множення над цілими числами.

Доведення. На кроках 2 та 4 множення на матриці $[K_0 \cdot K_1 \cdot C_{N+1.N+1}]$ та $[K_0 \cdot K_1 \cdot S_{N+1.N+1}]$ може бути замінено операціями бітового зсуву вліво, операціями додавання та віднімання за аналогією

з використанням ДКП при стисненні зображень. На кроці 4 множення на коефіцієнти $1/(2\cdot N\cdot(K_0\cdot K_1)\cdot K_2\cdot K_2)$ може бути замінено бітовими зсувами вправо, тому їх можна не враховувати при обчисленні кількості операцій множення.

Для обчислення кількості множень на кроці 3 використана властивість ДПФ (6). Тоді крок 3 потребує $N+1$ множень, два з яких не є комплексними множеннями. З урахуванням (1) та (6) кількість множень на кроці 3 буде $1+(N-1)\cdot 3+1 = 3N-1$. Теорема доведена.

Метод множення у вигляді Алгоритму 4 може бути цікавим у схемотехніці при реалізації багаторозрядної операції множення, так як метод має лінійну складність за кількістю операцій однорозрядного множення, а операції бітового зсуву, додавання та віднімання ефективно реалізуються на апаратному рівні.

Кількість коефіцієнтів для обчислення ДКП, ДСП. Для $N=2$ матриці $RC_{N+1,N+1}$ та $RS_{N+1,N+1}$ мають тільки один множник ± 1, не враховуючи знаки та нульові значення, тому не має потреби використовувати коефіцієнти $K_0 \cdot K_1 = 1$. Для $N=4$ матриці $[K_0 \cdot K_1 \cdot C_{5,5}]$ та $[K_0 \cdot K_1 \cdot S_{5,5}]$ мають 2 коефіцієнта ± 256, ± 181 при $K_0 \cdot K_1 = 256$, не враховуючи знаків та нульові значення. Продовжуючи для $N=8$, матриці $RC_{N+1,N+1}$ та $RS_{N+1,N+1}$ мають 4 коефіцієнта ± 512, ± 473, ± 362, ± 196 при $K_0 \cdot K_0 = 512$ і т. д. для більших N. Найбільший з множників буде степе-

ню двійки, тому коефіцієнти K_0, K_1 та K_2 треба підбирати таким чином, щоб кількість бітових зсувів була кратна довжині слова у бітах $8, 16, 32, 64, 128$.

Обчислення змінної розрядності для зменшення похибки заокруглення. Проаналізуємо збільшення довжини числа у бітах на протязі виконання алгоритму 4.

Будемо використовувати наступні позначення, що x та y – довжини у бітах одного розряду багаторозрядних чисел X_N та Y_N, відповідно; $k_0 + k_1$ – довжина у бітах коефіцієнта $K_0 \cdot K_1$ ($K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$) достатня, щоб мінімізувати похибки заокруглення при обчисленні елементів матриць $C_{N+1,N+1}$ і $S_{N+1,N+1}$, щоб зменшити вплив цих похибок на результат обчислення; n – довжина у бітах числа $N = 2^n$, що є кількістю розрядів чисел X_N та Y_N; $sign$ – кількість додаткових бітів для збереження знаку тимчасового числа, $complex$ – кількість додаткових бітів для збереження точності результату операції комплексного множення на кроці 3; res – кількість додаткових бітів для збереження результату різниці між дійсною та уявною частинами на кроці 5; w – довжина машинного слова у бітах 32, 64, 128 і т. д.

Розглянемо наступну лему.

Лема 2. При реалізації алгоритму 4 на всіх кроках його виконання повинно виконуватися наступне співвідношення:

$$x + y + k_0 + 3k_1 + 2k_2 + 3n + 3 + sign + complex + res \le w, \quad x + y \le k_0 + k_1. \tag{8}$$

Доведення. На кроці 1 довжина коефіцієнта $\pm(K_0 + K_1)$ у бітах дорівнює $sign + k_0 + k_1$, що відповідає максимальній довжині (у бітах) елементів у $N+1$ рядках і $N+1$ стовпчиках матриць $C_{N+1,N+1}$ і $S_{N+1,N+1}$. При обчисленні результату на кроці 2 довжина у бітах елементів у дійсній та уявній частинах вектор-стовпців $\hat{X}R_{N+1}$, $\hat{X}I_{N+1}$, $\hat{Y}R_{N+1}$, $\hat{Y}I_{N+1}$ буде $sign + k_1 + k_2 + x + n + 1$ та $sign + k_1 + k_2 + y + n + 1$, відповідно, враховуючи, що додавання $N+1$ елементів додає $n+1$ бітів у довжині результату. Після множення на кроці 3 довжина у бітах елементів у дійсній та уявній частинах вектор-стовпця $\hat{Z}_{2N}$ збільшується до $sign + 2(k_1 + k_2) + x + y + 2n + 2 + complex$, так як при множенні довжина результату у бітах є добутком довжин множників у бітах, і потрібно ще додатково враховувати операцію додавання (віднімання). На кроці 4 для обчислення кожного елемента вектор-стовпців ZR_{N+1}, ZI_{N+1} необхідно обчислити добуток $2N$ доданків однорозрядних множень чисел довжиною у $sign + 2(k_1 + k_2) + x + y + 2n + 2 + complex$ бітів на елементи довжиною у $k_0 + k_1$ бітів. По завершенню кроку 4 довжина кожного елемента вектор-стовпців ZR_{N+1}, ZI_{N+1} повинна мати наступну кількість бітів для врахування похибок заокруглення при обчисленні у полі цілих чисел на всіх кроках:

$$x + y + 2(k_1 + k_2) + k_0 + k_1 + 3n + 3 + sign + complex + res \le w.$$

Лема доведена.

З виразу (8) видно, що навіть при невеликих значеннях x, y та n, довжини слова у 32 бітів буде недостатньо для врахування похибок заокруглення при обчисленні.

Вибір довжини слова у бітах. Розглянемо (8) при $sign = complex = res = 1$, $x = y = 4$, $k_0 = k_2 = 0$, $k_1 > 0$ та $w = 32$. Тоді (8) можна записати у наступному вигляді:

$$4 + 4 + 3k_1 + 3n + 3 + 1 + 1 + 1 \le 32. \tag{9}$$

У лівій та правій частинах можна зробити скорочення $3k_1 + 3n \le 18$. Розділимо на число 3 ліву та праву частини та отримаємо, що $k_1 + n \le 6$. При множенні двох чисел розрядністю $N = 2^{n=2}$ отримаємо, що $k_1 \le 4$. Значення π та тригонометричні функції обчислюються з подвійною точністю, тому в (9) довжини 32 (у бітах) недостатньо для врахування похибки заокруглення на всіх

кроках алгоритму. Необхідно, щоб $8 \le k_1$ при $x = y = 4$, $k_0 = k_2 = 0$ для отримання коректного результату, що сумарно перевищує довжину слова у 32 біти. Тому для реалізації треба використовувати щонайменше 64-бітні слова. Але на кроках 2, 3, 4 збільшується довжина слів (у розрядах), що збільшує складність за кількістю однорозрядних операцій і зменшує ефективність Алгоритму 4. Для того, щоб вирішити це протиріччя необхідно використовувати коефіцієнти $k_0 \ge 0$ або $k_2 \ge 0$.

Таблиці коефіцієнтів для врахування похибки заокруглення. Далі наведено табл. 1 – 4 залежності коефіцієнтів $K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$, $K_2 = 2^{k_2}$ від кількості розрядів N числа та довжини розрядів x та y (у бітах).

ТАБЛИЦЯ 1. Залежність мінімальних довжин k_0, k_1, k_2 (у бітах) коефіцієнтів $K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$, $K_2 = 2^{k_2}$, де k_0, k_1, k_2 – цілі числа, від кількості розрядів $N = 2^n$, де $n = \overline{2,9}$, при $x = y = 4$ – довжина у бітах одного розряду

$n=2$ $N=4$ k_0,k_1,k_2	$n=3$ $N=8$ k_0,k_1,k_2	$n=4$ $N=16$ k_0,k_1,k_2	$n=5$ $N=32$ k_0,k_1,k_2	$n=6$ $N=64$ k_0,k_1,k_2	$n=7$ $N=128$ k_0,k_1,k_2	$n=8$ $N=256$ k_0,k_1,k_2	$n=9$ $N=512$ k_0,k_1,k_2
0 – 8 – 0	0 – 9 – 0	0 – 11 – 0	0 – 13 – 0	0 – 13 – 0	0 – 14 – 0	0 – 15 – 0	0 – 16 – 0
1 – 7 – 0	1 – 8 – 0	1 – 10 – 0	1 – 12 – 0	1 – 12 – 0	1 – 13 – 0	1 – 14 – 0	1 – 15 – 0
2 – 6 – 0	2 – 7 – 0	2 – 9 – 0	2 – 11 – 0	2 – 11 – 0	2 – 12 – 0	2 – 13 – 0	2 – 14 – 0
3 – 5 – 0	3 – 6 – 0	3 – 8 – 0	3 – 10 – 0	3 – 10 – 0	3 – 11 – 0	3 – 12 – 0	3 – 13 – 0
4 – 4 – 1	4 – 5 – 0	4 – 7 – 0	4 – 9 – 0	4 – 9 – 0	4 – 10 – 0	4 – 11 – 0	4 – 12 – 0
5 – 3 – 1	5 – 4 – 0	5 – 6 – 0	5 – 8 – 0	5 – 8 – 0	5 – 9 – 0	5 – 10 – 0	5 – 11 – 0
6 – 2 – 3	6 – 3 – 1	6 – 5 – 0	6 – 7 – 0	6 – 7 – 0	6 – 8 – 0	6 – 9 – 0	6 – 10 – 0
7 – 1 – 4	7 – 2 – 2	7 – 5 – 1	7 – 6 – 0	7 – 6 – 0	7 – 7 – 0	7 – 8 – 0	7 – 9 – 0
8 – 0 – 5	8 – 1 – 3	8 – 4 – 0	8 – 5 – 0	8 – 5 – 1	8 – 6 – 0	8 – 7 – 0	8 – 8 – 0
	9 – 0 –4	9 – 3 – 1	9 – 4 – 1	9 – 4 – 2	9 – 5 – 0	9 – 6 – 0	9 – 7 – 0
		10 – 2 – 2	10 – 3 – 2	10 – 3 – 3	10 – 4 – 0	10 – 5 – 0	10 – 6 – 0
		11 – 1 – 3	11 – 2 – 3	11 – 2 – 4	11 – 3 – 1	11 – 4 – 1	11 – 5 – 1
		12 – 0 – 4	12 – 1 – 4	12 – 1 – 5	12 – 2 – 2	12 – 3 – 2	12 – 4 – 2
			13–0 – 5	13 – 0 – 6	13 – 1 – 3	13 – 2 – 3	13 – 3 – 3
					14 – 0 – 4	14 – 1 – 4	14 – 2 – 4
						15 – 0 – 5	15 – 1 – 5
							16 – 0 – 5

ТАБЛИЦЯ 2. Залежність мінімальної довжини $k_0 + k_1$ (у бітах) коефіцієнта $K_0 \cdot K_1$ ($K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$), де k_0, k_1 – цілі числа, від кількості розрядів $N = 2^n$, де $n = \overline{10,16}$, при $x = y = 4$ – довжина у бітах одного розряду

$n=10$ $N=1024$ k_0+k_1	$n=11$ $N=2048$ k_0+k_1	$n=12$ $N=4096$ k_0+k_1	$n=13$ $N=8192$ k_0+k_1	$n=14$ $N=16384$ k_0+k_1	$n=15$ $N=32768$ k_0+k_1	$n=16$ $N=65536$ k_0+k_1
17	18	19	20	21	22	23 (14+9)

ТАБЛИЦЯ 3. Залежність мінімальної довжини $k_0 + k_1$ (у бітах) коефіцієнта $K_0 \cdot K_1$ ($K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$), де k_0, k_1 – цілі числа, від кількості розрядів $N = 2^n$, де $n = \overline{2,10}$; при $x = y = 8$ – довжина у бітах одного розряду

$n=2$ $N=4$ k_0+k_1	$n=3$ $N=8$ k_0+k_1	$n=4$ $N=16$ k_0+k_1	$n=5$ $N=32$ k_0+k_1	$n=6$ $N=64$ k_0+k_1	$n=7$ $N=128$ k_0+k_1	$n=8$ $N=256$ k_0+k_1	$n=9$ $N=512$ k_0+k_1	$n=10$ $N=1024$ k_0+k_1
16	19	20	21	22	22	23	24	25

ТАБЛИЦЯ 4. Залежність мінімальної довжини $k_0 + k_1$ (у бітах) коефіцієнта $K_0 \cdot K_1$ ($K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$), де k_0, k_1 – цілі числа, від кількості розрядів $N = 2^n$, де $n = \overline{2,4}$, при $x = y = 12$ – довжина у бітах одного розряду

$n=2$ $N=4$ k_0+k_1	$n=3$ $N=8$ k_0+k_1	$n=4$ $N=16$ k_0+k_1
26	28	28

З таблиць можна побачити, що існує найменша довжина у бітах $k_0 + k_1$, починаючи з якої похибка заокруглення не впливає на результат обчислення. З табл. 1 бачимо, що при збільшенні k_0, починаючи з певного значення необхідно збільшувати k_2 для компенсації похибки заокруглення. Дослідження широкого діапазону значень $k_0 + k_1$ дозволяє використовувати однакові реалізації алгоритмів при різних значеннях аргументу $N = 2^n$.

Враховуючи взаємозв'язок $k_0 + k_1 = B_{x,y,n}$, де $B_{x,y,n}$ – найменша кількість бітів при заданих x, y, n, при збільшенні k_1 точність обчислення збільшується при остаточному заокругленні результату множення. Взагалі, у більшості випадків збільшення одного зі значень k_0, k_1, k_2 покращує точність результату.

Порівняльна таблиця за кількістю операцій однорозрядного множення. У діапазоні до 4096 бітів «діагональний метод» за рахунок максимального використання кешу ядра процесору випереджає всі відомі методи. Будемо вважати, що при виконанні Алгоритму 4 велике число розбивається на розряди, кожному з яких відповідає байт. При виконанні алгоритму, який реалізує метод Карацуби велике число розбивається на розряди, кожному з яких відповідає подвійне слово (4 байти). У обох алгоритмах відбуваються множення 32-бітних слів. Вважаємо, що у Алгоритмі 4 перед множенням для вирівнювання довжини у 32 біти до розрядів додаються нульові біти. Як видно із табл. 5 запропонований Алгоритм 4 починає випереджати метод Карацуби за кількістю операцій однорозрядного (32-бітного) множення при множенні чисел довжиною у 4096 бітів.

ТАБЛИЦЯ 5. Залежність кількості операцій 32-бітного множення від довжини числа (у бітах) при реалізації Алгоритму 4 та методу Карацуби

Алгоритм 4			Метод Карацуби		
Кількість розрядів, $N_{x=8}$	Довжина числа у бітах, $N_{x=8} \cdot x$	Кількість множень, $3N-1$	Кількість розрядів, $N_{x=32}$	Довжина числа у бітах, $N_{x=32} \cdot x$	Кількість множень, $3^{\log_2 N}$
4	32	11	4	128	9
8	64	23	8	256	27
16	128	47	16	512	81
32	256	95	32	1024	243
64	512	191	64	2048	729
128	1024	383	128	4096	2187
256	2048	767	256	8192	6561
512	4096	1535	512	16384	19683
1024	8192	3071	1024	32768	59049

Немає потреби порівнювати з методом на основі ШПФ, так як Алгоритм 4 є однією з його модифікацій з ще більшою оптимізацією за кількістю операцій однорозрядного множення.

Висновки. У роботі запропоновано метод багаторозрядного множення на основі ДКП та ДСП, що дозволяє розділити обчислення для дійсної та уявної частин ДПФ дійсного сигналу. Наведено алгоритм 2 обчислення ДПФ у полі дійсних чисел, який виконує на 33 % менше операцій однорозрядного множення чисел з плаваючою комою у порівнянні з Алгоритмом 1, який виконує обчислення ДПФ з комплексними числами. Алгоритм 3 побудовано на основі ДКП та ДСП, що дозволяє виключити операції з комплексними числами при обчисленні ДПФ та ОДПФ. В Алгоритмі 4 на кожному кроці відбуваються обчислення з цілими числами за рахунок додаткових побітових зсувів та заокруглень. У вигляді теореми доведено, що реалізація операції багаторозрядного множення у вигляді Алгоритму 4 має лінійну складність за кількістю операцій однорозрядного множення цілих чисел. Наведено приклади обчислення для кожного алгоритму. Надано аналіз вибору довжини слова при обчислені Алгоритму 4. Надані таблиці залежності мінімальних довжин коефіцієнтів $K_0 = 2^{k_0}$, $K_1 = 2^{k_1}$, $K_2 = 2^{k_2}$, де k_0, k_1, k_2 – цілі числа, від довжини багаторозрядного числа та довжини розряду (у бітах). На основі порівняльного аналізу показано, що запропонований метод множення на основі ДКП та ДСП у полі цілих чисел починає випереджати метод Карацуби за кількістю 32-бітних операцій множення при множенні чисел, починаючи з довжини у 4096 бітів.

Список літератури

1. Анісімов А.В. Алгоритмічна теорія великих чисел. *Модулярна арифметика великих чисел*. Київ: Академперіодика, 2001. 153 с. http://books.zntu.edu.ua/book_info.pl?id=21106
2. Задірака В., Олексюк О. Комп'ютерна арифметика багаторозрядних чисел. Київ: Наукове видання, 2003. 263 с.
3. Задирака В.К. Теория вычисления преобразования Фурье. Киев: Наук. думка, 1983. 213 с.
4. Задірака В.К., Терещенко А.М. Комп'ютерна арифметика багаторозрядних чисел у послідовній та паралельній моделях обчислень. Киев: Наук. думка, 2021. 136 с.

5. Качко Е.Г. Распараллеливание алгоритмов умножения чисел многократной точности. *Вестник Уфимского государственного авиационного технического университета*. 2011. **E 15**, № 5 (45). С. 142–147. http://omega.sp.susu.ru/books/conference/PaVT2011/short/015.pdf
6. 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. N 806. С. 195–199. http://nbuv.gov.ua/UJRN/VNULPKSM_2014_806_31
7. Хіміч О.М., Сидорук В.А. Використання мішаної розрядності у математичному моделюванні. Математичне та комп'ютерне моделювання. *Серія: Фізико-математичні науки. Зб. наук. праць*. 2019. 19. С. 180–187. https://doi.org/10.32626/2308-5878.2019-19.180-187
8. Карацуба А.А., Офман Ю.П. Умножение многоразрядных чисел на автоматах. *ДАН СССР*. 1962. **145** (2). С. 293–294. http://mi.mathnet.ru/dan26729
9. Cooley J.W., Tukey J.W. An algorithm for the machine calculation of complex Fourier Series. *Math Compt*. 1965. **19**. P. 257–301. https://doi.org/10.1090/S0025-5718-1965-0178586-1
10. Schonhage A., Strassen V. Schnelle Multiplikation großen Zahlen. *Computing*. 1971. **7** (3–4). P. 281–292. https://doi.org/10.1007/BF02242355
11. Pitassi D.A. Fast convolution using the Walsh transform. *Applications of Walsh Functions*. April, 1971. P. 130–133. https://doi.org/10.1109/TEMC.1971.303141
12. Терещенко А.Н., Мельникова С.С., Гнатив Л.А., Задирака В.К., Кошкина Н.В. Реализация операции умножения с использованием преобразования Уолша. *Проблемы управления и информатики*. 2010. № 2. С. 102–126.
13. Montgomery P.L. Modular Multiplication Without Trial Division. *Math. Comp*. 1985. **44**. P. 519–521. https://doi.org/10.1090/S0025-5718-1985-0777282-X
14. 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: 14.12.2021)
15. Ahmed N., Natarajan T., Rao K.R. Discrete cosine transform. *IEEE Trans. Comput*. 1974. **C–23** (1). P. 90–93. https://doi.org/10.1109/T-C.1974.223784
16. Jain A.K. A fast Karhunen–Loève transform for a class of random processes. *IEEE Trans. on Communications*. 1976. 24. 1023. https://doi.org/10.1109/TCOM.1976.1093409
17. Gluth R. Regular FFT–related transform kernels for DCT/DST – based polyphase filter banks. *Proc. IEEE ICASSP*. Toronto, Canada, May 1991. P. 2205–2208. https://doi.org/10.1109/ICASSP.1991.150852
18. Чичева М.А. Эффективный алгоритм дискретного косинусного преобразования четной длины. *Компьютерная оптика*. 1998. №. 18. С. 147–149. https://doi.org/10.1080/713696822
19. Гнатів Л.О., Луц В.К. Цілочислові модифіковані синус-косинусні перетворення типу VII. Метод побудови і роздільні направлені адаптивні перетворення для intra-прогнозування з блоками яскравості 8 x 8 у кодуванні зображень/відео. *Кібернетика та системний аналіз*. 2021. Т. 57, № 1. С. 178–190.

Одержано 14.12.2021

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

Задірака Валерій Костянтинович,
академік НАН України, доктор фізико-математичних наук, керівник відділу
Інституту кібернетики імені В.М. Глушкова НАН України, Київ.
https://orcid.org/0000-0001-9628-0454

MSC 90C15, 49M27

Andrii Tereshchenko [*], Valeriy Zadiraka

Implementation of Multidigit Multiplication Basing on Discrete Cosine and Sine Transforms

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

Introduction. The emergence of new parallel computing systems such as multi-core processors, clusters, distributed systems, due to the solution of various applications in different spheres. Among such problems are the calculation of systems of linear algebraic equations with the number of unknown 33-35 million, the calculation of nuclear reactor shells, modeling of physical and chemical processes, aerodynamics, hydrodynamics, information security, and so on. This greatly expands the use of multidigit arithmetic, due to the fact that ignoring rounding errors leads to the fact that sometimes computer solutions are obtained that do not correspond to the physical content. Multidigit multiplication operation is an integral part of the exponentiation by module operation, the speed of which determines the speed of asymmetric cryptographic software and hardware complexes.

This paper presents algorithms for implementing the multiplication operation of two *N*-digit numbers based on discrete cosine and sine transforms (DCT and DST) by separating the calculation for the real and imaginary parts of the DFT. Calculation of DCT and DST at the expense of additional bit shifts, additions and subtractions reduces the algorithm complexity to linear complexity by the number of integer multiplication operations.

The purpose of the article is to reduce the number of multiplication operations to speed up the execution time of the multiplication operation of two *N*-bit numbers based on discrete transforms. Reduce the number of complex multiplication operations. Reduce the overall computational complexity and find a modification in which the calculation steps will correspond to DCT, DSP, IDCT and IDST. Use the coefficients to take into account the rounding errors to exclude multiplication operations on calculating DCT, DST, IDCT and IDST.

Results. The relationship between DCT, DST and DFT of a real signal is considered, which allows to separate calculations for real and imaginary parts of DFT of real signals. The computational complexity is reduced almost twice at the expense of use of DFT properties of real signals. It is shown that after optimization steps of the algorithm calculation correspond to DCT, DST, IDCT and IDST. Using additional coefficients, which allow to take into account rounding errors at each step so that all calculations use integers. An analysis of the choice of word length in the calculation is given. For each algorithm, examples of calculation are given. Tables of dependence of the minimum lengths of the coefficients on the length of the multidigit number and the length of the digit (in bits) are given.

Conclusions. Multiplication algorithms of two *N*-digit numbers based on discrete cosine and sine transforms (DCT and DST) are presented in this paper. Separating the calculation for the real and imaginary parts of the DFT allows to reduce the number of multiplication operations by 33%. The use of additional coefficients and calculation of DCT, DST, IDCT, IDST at the expense of bit shifts, additions and subtractions reduces the complexity of the multiplication algorithm of two *N*-digit numbers to linear complexity by the number of simple integer multiplication operations. Based on comparative analysis, it is shown that the proposed method of multiplication based on DCT and DST using integers begins to exceed the Karatsuba method by the number of 32-bit multiplication operations when multiplying numbers, starting with a length of 4096 bits.

Keywords: multidigit multiplication, multidigit arithmetic, asymmetric cryptography, discrete cosine transform, discrete sine transform, discrete Fourier transform, fast algorithm for Fourier calculation.