УДК 519.61:004.272 https://doi.org/10.15407/pp2026.01.102

Г. І. Малашонок, С. С. Сухарський

БЛОКОВО-РЕКУРСИВНИЙ ПІДХІД ДО УНІТАРНОГО РОЗКЛАДУ МАТРИЦЬ

Сингулярне розкладання матриці є одним з основних інструментів чисельної лінійної алгебри та широко застосовується у задачах обробки даних, оптимізації, комп'ютерного зору та машинного навчання. Класичний підхід до його обчислення базується на алгоритмі Хаусхолдера, який дозволяє звести довільну матрицю до тридіагональної або бідіагональної форми з подальшим обчисленням сингулярних значень. Однак зі зростанням розмірності матриць виникають суттєві труднощі з ефективним розпаралелюванням таких алгоритмів, що обмежує їхню продуктивність на сучасних високопродуктивних обчислювальних системах.

Одним із перспективних підходів до подолання цих обмежень є використання блоково-рекурсивних алгоритмів. Такі алгоритми дозволяють розділяти початкову задачу на незалежні блокові підзадачі, що можуть виконуватися паралельно, забезпечуючи високий ступінь масштабованості на багатопроцесорних системах та кластерах із розподіленою пам'яттю.

У даній роботі запропоновано новий алгоритм ортогонального (унітарного) блоково-рекурсивного розкладу симетричних матриць, який базується на використанні QR- та QP-розкладів. Алгоритм дозволяє побудувати ортогональні (унітарні) множники та отримати стрічкову матрицю, що зменшує обчислювальну складність подальших етапів спектрального розкладу.

Отримано оцінку обчислювальної складності алгоритму та показано, що її асимптотичний порядок визначається складністю множення матриць. Експериментальні дослідження продуктивності проведено у середовищі паралельних обчислень із використанням прискорення матричних операцій на графічному процесорі. Отримані результати демонструють ефективність запропонованого алгоритму при зростанні розміру задачі та підтверджують узгодженість експериментальної складності з теоретичними оцінками. Значення похибки реконструкції матриці знаходяться на рівні машинної точності для чисел подвійної точності, що свідчить про чисельну стабільність запропонованого підходу.

Ключові слова: унітарний розклад матриці, блоково-рекурсивний алгоритм, паралельні обчислення, графічний процесор

G. I. Malaschonok, S. S. Sukharskyi

A BLOCK-RECURSIVE APPROACH TO UNITARY MATRIX DECOMPOSITION

Singular value decomposition is one of the fundamental tools of numerical linear algebra and is widely used in data processing, optimization, computer vision, and machine learning. The classical approach to its computation is based on the Householder algorithm, which reduces an arbitrary matrix to a tridiagonal or bidiagonal form followed by the computation of singular values. However, as the matrix size increases, significant difficulties arise in efficiently parallelizing such algorithms, which limits their performance on modern high-performance computing systems.

One promising approach to overcoming these limitations is the use of block-recursive algorithms. Such algorithms allow the original problem to be decomposed into independent block subproblems that can be processed in parallel, providing a high degree of scalability on multiprocessor systems and distributed-memory clusters.

This paper proposes a new block-recursive algorithm for the orthogonal (or unitary) decomposition of symmetric matrices based on QR and QP decompositions. The algorithm constructs orthogonal (unitary) factors and produces a band matrix, reducing the computational complexity of subsequent stages of spectral decomposition.

A complexity analysis is presented showing that the asymptotic order of the algorithm is determined by the complexity of matrix multiplication.Experimental performance studies were carried out in a parallel computing environment using GPU acceleration for matrix operations. The obtained results demonstrate the efficiency of the proposed algorithm as the problem size increases and confirm the agreement between the experimental complexity and the theoretical estimates. The reconstruction error of the matrix remains at the level of machine precision for double-precision floating-point numbers, which indicates the numerical stability of the proposed approach.

Keywords: unitary matrix decomposition, block-recursive algorithm, parallel computing, graphics processing unit

Вступ

Задача ортогонального (унітарного) розкладу симетричної матриці до тридіагональної або стрічкової форми є однією з базових задач чисельної лінійної алгебри. Вона виникає у багатьох прикладних задачах, зокрема, при знаходженні власних значень матриць, розв'язуванні задач спектрального аналізу, моделюванні фізичних процесів та обробці великих масивів даних. У більшості сучасних алгоритмів обчислення власних значень попереднє перетворення матриці до тридіагонального або стрічкового вигляду є необхідним етапом, що значною мірою визначає загальну обчислювальну складність задачі. Зі зростанням розмірів задач та розвитком високопродуктивних обчислювальних систем особливого значення набуває розробка алгоритмів, які дозволяють ефективно використовувати паралельні архітектури. Традиційні методи розкладу матриць, хоча й забезпечують високу чисельну стабільність, часто мають обмежені можливості для паралелізації. У зв'язку з цим актуальною є задача розробки нових алгоритмічних підходів, що поєднують чисельну стійкість класичних методів із високою ефективністю паралельного виконання. Одним із перспективних напрямів розв'язання цієї задачі є використання блоково-рекурсивних алгоритмів, які дозволяють природно організувати паралельні обчислення та ефективно використовувати сучасні високопродуктивні обчислювальні системи.

Задача тридіагоналізації матриці тісно пов'язана з класичними методами QR-розкладу, які широко застосовуються у чисельній лінійній алгебрі завдяки своїй чисельній стійкості. Основні підходи до побудови QR-розкладу — обертання Гівенса, ортогоналізація Грама–Шмідта та відбиття Хаусхолдера — детально описані у класичних роботах Голуба і Ван Лоана [1], а також Трефезена і Бау [2]. Одним із важливих напрямів розвитку алгоритмів чисельної лінійної алгебри стало використання методів швидкого множення матриць. У роботі Штрассена показано можливість зниження складності множення матриць до $\boldsymbol{O}\left(\boldsymbol{n}^{\log_2 7}\right)$ [3], а подальші дослідження Копперсміта і Віноградa дозволили зменшити асимптотичну оцінку до $\boldsymbol{O}(\boldsymbol{n}^{2.38})$ [4]. Ці результати створили передумови для розробки ефективних блоково-рекурсивних алгоритмів у задачах чисельної лінійної алгебри.

Однією з ранніх робіт у цьому напрямі є дослідження Шонхаге [5], у якому розглянуто швидке виконання унітарних перетворень великих матриць. У роботі показано, що такі перетворення можуть бути організовані таким чином, що їхня асимптотична складність має той самий порядок, що і складність множення матриць. Подальший розвиток цієї ідеї привів до створення рекурсивних алгоритмів факторизації матриць. Зокрема, у роботах [6, 7, 8] розглядалось застосування блоково-рекурсивних алгоритмів до задач унітарного розкладу матриць, пов'язаних із задачею QR-розкладу.

Важливим результатом цього напрямку стала робота Деммела та співавторів [9], у якій доведено, що швидкі алгоритми множення матриць можуть бути використані для розв'язання стандартних задач чисельної лінійної алгебри (SVD, LU, QR) із тією самою асимптотичною складністю, зберігаючи водночас чисельну стійкість. Автори показали, що навіть у разі використання так званих слабо стійких алгоритмів множення матриць забезпечується зворотна стійкість відповідних матричних розкладів.

Подальші дослідження показали, що рекурсивні матричні алгоритми можуть ефективно масштабуватися у паралельних обчислювальних системах із динамічним розподілом обчислювальних ресурсів [6]. Для реалізації таких алгоритмів розроблено середовище виконання DAP, яке забезпечує ефективне розпаралелювання рекурсивних алгоритмів у системах із розподіленою пам'яттю [10, 11]. Це створює передумови для розробки нових блоково-рекурсивних алгоритмів.

Метою даної роботи є розробка та дослідження ефективного алгоритму ортогонального (унітарного) розкладу симетричних матриць до стрічкової форми на основі блоково-рекурсивних QR та QP пере-

творень. Для досягнення цієї мети у роботі запропоновано алгоритм Q3RP, що базується на послідовному застосуванні QR та QP перетворень. Також проведено аналіз обчислювальної складності алгоритму та експериментальне дослідження його продуктивності у паралельному середовищі виконання DAP із використанням прискорення матричних операцій на графічному процесорі.

Ортогональний розклад симетричної матриці

Ортогональний (унітарний) розклад симетричної матриці до тридіагональної форми є одним із ключових етапів у багатьох алгоритмах знаходження власних значень. Класичні методи такого розкладу, зокрема, метод відбиттів Хаусхолдера, мають складність порядку $\boldsymbol{O(n^3)}$ і реалізуються переважно як послідовні алгоритми, що обмежує їхню ефективність у паралельних обчислювальних системах [1, 2]. У зв'язку з цим значний інтерес становлять блоково-рекурсивні алгоритми, які дозволяють краще використовувати паралельну архітектуру сучасних обчислювальних платформ.

QR-розклад. Одним із таких підходів є блоково-рекурсивні алгоритми, що базуються на QR-розкладі матриці. QR-розклад представляє довільну матрицю $\boldsymbol{A}$ у вигляді добутку ортогональної матриці Q та верхньої трикутної матриці $\boldsymbol{R}$:

$$\boldsymbol{A = Q^T R}, \quad \boldsymbol{Q^T Q = Q Q^T = I}, \tag{1}$$

де $\boldsymbol{Q}$ — ортогональна матриця, а $\boldsymbol{R}$ — верхня трикутна матриця.

Одним із поширених способів побудови такого розкладу є метод обертань Ґівенса, який послідовно занулює піддіагональні елементи матриці за допомогою ортогональних матриць повороту. Для двох елементів вектора $\boldsymbol{(a, b)^T}$ відповідна матриця повороту має вигляд

$$\boldsymbol{g(\alpha, \gamma)} = \begin{pmatrix} \boldsymbol{\alpha} & \boldsymbol{\gamma} \\ \boldsymbol{-\gamma} & \boldsymbol{\alpha} \end{pmatrix}, \quad \boldsymbol{\alpha^2 + \gamma^2 = 1}, \tag{2}$$

що забезпечує ортогональність перетворення. Узагальненням такого перетворення для матриці є матриця Ґівенса

$$\boldsymbol{G(i, j)} = \mathrm{diag}(\boldsymbol{I, g(\alpha, \gamma), I}), \tag{3}$$

яка відрізняється від одиничної лише у двох рядках. Завдяки цьому відповідні перетворення є локальними та придатними для паралельної реалізації. Для щільної матриці розміру $\boldsymbol{n \times n}$ обчислювальна складність такого алгоритму становить приблизно $\boldsymbol{2n^3}$ арифметичних операцій.

Подальший розвиток цієї ідеї привів до появи блоково-рекурсивних алгоритмів QR-розкладу. У роботі [6] запропоновано алгоритм, у якому матриця

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix} \tag{4}$$

розбивається на чотири блоки однакового розміру. Алгоритм складається з трьох основних етапів.

На першому етапі виконується QR-розклад блоку $\boldsymbol{C}$

$$\boldsymbol{Q_1 C = C^U}, \tag{5}$$

у результаті чого формується верхньотрикутна матриця $\boldsymbol{C^U}$ та ортогональне перетворення $\boldsymbol{Q_1}$. Отримане перетворення застосовується до всієї матриці.

Другим етапом є спеціальна процедура, яка називається QP-розкладом – Рис. 1. Вона використовується для занулення елементів області, утвореної нижньою трикутною частиною блоку $\boldsymbol{A}$ та верхньою трикутною частиною блоку $\boldsymbol{C^U}$. Ця область має форму так званого «паралелограму». У процесі QP-розкладу будується ортогональне перетворення $\boldsymbol{Q_2}$, яке перетворює матрицю

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{C^U} \end{pmatrix} \tag{6}$$

до верхньотрикутного вигляду.

У результаті застосування перетворення $\boldsymbol{Q_2}$ матриця $\boldsymbol{M_1}$ набуває вигляду

$$\boldsymbol{M_2 = Q_2} \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C^U} & \boldsymbol{D_1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{A^U} & \boldsymbol{B_1} \\ \boldsymbol{0} & \boldsymbol{D_2} \end{pmatrix}, \tag{7}$$

де $\boldsymbol{A^U}$ — верхньотрикутний блок, $\boldsymbol{B_1}$ — оновлений правий блок, а $\boldsymbol{D_2}$ — модифікований нижній блок матриці.

На третьому етапі виконується QR-розклад оновленого блоку $\boldsymbol{D_2}$, що приводить до загального розкладу

$$\boldsymbol{M} = \boldsymbol{Q}\,\boldsymbol{R}, \tag{8}$$

де

$$\boldsymbol{Q} = \mathrm{diag}(\boldsymbol{I}, \boldsymbol{Q_3})\,\boldsymbol{Q_2}\,\mathrm{diag}(\boldsymbol{I}, \boldsymbol{Q_1}). \tag{9}$$

QP-розклад. QP-розклад будується за блоково-рекурсивним принципом. Ми шукаємо розклад матриці $\boldsymbol{P}$:

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{C^U} \end{pmatrix}$$

$$Q_2 P = P^U, \quad P = Q_2^T P^U, \quad Q_2^{-1} = Q_2^T. \tag{10}$$

Матриця P є лівою половиною матриці M_1. Ми розбили кожен із двох блоків, що утворюють матрицю P (розміру $2n \times n$), на чотири рівні частини. У результаті отримано 8 блоків, включаючи один нульовий блок і два верхньотрикутні блоки f^U та h^U. Необхідно занулити всі елементи матриці P, розташовані між верхньою та нижньою діагоналями, включно із нижньою діагоналлю. Нас цікавить блокова процедура. Оскільки n є парним, паралелограм, утворений цими діагоналями, можна розділити на 4 частини за допомогою двох середніх ліній. У результаті отримуємо 4 однакові паралелограми. Для розкладу кожного з них процедура QP викликається рекурсивно 4 рази. Обробка виконується у такому порядку: спочатку нижній лівий паралелограм (P_{ll}), потім одночасно зануляються верхній лівий (P_{ul}) та нижній правий (P_{lr}), і на завершення виконується розклад паралелограма у верхньому правому куті (P_{ur}). Відповідні ортогональні матриці розміру $n \times n$ позначаються Q_{ll}, Q_{ul}, Q_{lr}таQ_{ur}.

$$P = (AC^U) = \begin{pmatrix} a & c \\ b & d \\ f^U & g \\ 0 & h^U \end{pmatrix},$$

$$P^U = (A^U 0) = \begin{pmatrix} a^U & c_1 \\ 0 & d^U \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

$$P_{ll} = Q_{ll}\begin{pmatrix} b & d \\ f^U & g \end{pmatrix} = \begin{pmatrix} b^U & d_1 \\ 0 & g_1 \end{pmatrix},$$

$$P_{ul} = Q_{ul}\begin{pmatrix} a & c \\ b^U & d_1 \end{pmatrix} = \begin{pmatrix} a^U & c_1 \\ 0 & d_2 \end{pmatrix},$$

$$P_{lr} = Q_{lr}\begin{pmatrix} 0 & g_1 \\ 0 & h^U \end{pmatrix} = \begin{pmatrix} 0 & g^U \\ 0 & 0 \end{pmatrix},$$

$$P_{ur} = Q_{ur}\begin{pmatrix} 0 & d_2 \\ 0 & g^U \end{pmatrix} = \begin{pmatrix} 0 & d^U \\ 0 & 0 \end{pmatrix}. \tag{11}$$

У результаті отримуємо матриці Q_2 та P^U:

$$Q_2 = \overline{Q_{ur}}\,\overline{Q_2}\,\overline{Q_{ll}}, \quad Q_2 P = P^U, \tag{12}$$

де

$$\overline{Q_{ur}} = \mathrm{diag}\left(I_{\frac{n}{2}}, Q_{ur}, I_{\frac{n}{2}}\right),$$
$$\overline{Q_2} = \mathrm{diag}(Q_{ul}, Q_{lr}),$$
$$\overline{Q_{ll}} = \mathrm{diag}\left(I_{\frac{n}{2}}, Q_{ll}, I_{\frac{n}{2}}\right). \tag{13}$$

Схему блоково-рекурсивного QR-розкладу представлено на Рис. 1.

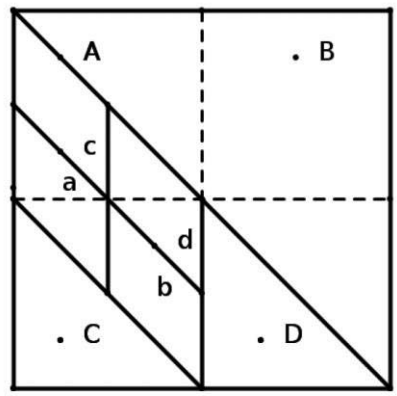

Рис. 1. Схема блоково-рекурсивного QR-розкладу

Складність QP-розкладу можна оцінити, підрахувавши кількість множень матричних блоків. У процесі виконання алгоритму необхідно виконати 28 множень блоків розміру $\boldsymbol{n/2} \times \boldsymbol{n/2}$: 8 множень використовуються у процесі обчислення матриць $\boldsymbol{P_{ll}}$ та $\boldsymbol{P_{ul}}$, а ще 20 — при побудові матриці $\boldsymbol{Q_2}$. Тому загальна кількість операцій задовольняє рекурентному співвідношенню

$$\boldsymbol{C_{qp}(2n) = 4C_{qp}(n) + 28M(n/2)}, \tag{14}$$

де $\boldsymbol{M(n)}$ — складність множення матриць розміру $\boldsymbol{n \times n}$.

Нехай складність множення двох матриць розміру $\boldsymbol{n \times n}$ дорівнює

$$\boldsymbol{M(n) = \gamma n^{\beta}},$$

(15)

а $\boldsymbol{n = 2^k}$. Тоді рекурентне співвідношення набуває вигляду

$$\boldsymbol{C_{qp}(2^{k+1}) = 4C_{qp}(2^k) + 28M(2^{k-1})}.$$

(16)

Розгортаючи рекурсію, отримуємо

$$\boldsymbol{C_{qp}(2^{k+1}) = 4^k C_{qp}(2) + 28\gamma \sum_{i=0}^{k-1} 4^{k-i-1} 2^{i\beta}}.$$

(17)

Для розв'язання рекурентного співвідношення врахуємо базовий випадок

$$\boldsymbol{C_{qp}(2) = 6}.$$

(18)

Тоді отримуємо явний вираз для складності QP-розкладу

$$\boldsymbol{C_{qp}(n) = \frac{28\gamma(n^{\beta} - n^2)}{2^{\beta}(2^{\beta} - 4)} + 6n^2}.$$

(19)

Обмежуючись старшим ступенем за n, отримуємо асимптотичну оцінку

$$\boldsymbol{C_{qp}(n) = \varphi(\beta)\gamma n^{\beta}, \quad \varphi(\beta) = \frac{28}{2^{\beta}(2^{\beta} - 4)}}.$$

$$\boldsymbol{\varphi(3) = \frac{7}{8}, \quad \varphi(\log_2 7) = \frac{4}{3}}.$$

(20)

Складність блоково-рекурсивного QR-розкладу визначається з урахуванням того, що алгоритм містить два рекурсивні виклики QR-процедури, одну процедуру QP-розкладу та кілька операцій множення матриць. Якщо позначити складність QP-розкладу як

$$\boldsymbol{C_{qp}(n) = \alpha n^{\beta}},$$

(21)

а складність множення матриць як (15), то кількість операцій QR-розкладу задовольняє рекурентному співвідношенню

$$\boldsymbol{C_{qr}(n) = 2C_{qr}(n/2) + C_{qp}(n/2) + 8M(n/2)}$$

(22)

Нехай $\boldsymbol{n = 2^k}$. Тоді

$$\begin{aligned}\boldsymbol{C_{qr}(2^k)} &= \boldsymbol{2C_{qr}(2^{k-1}) + C_{qp}(2^{k-1}) + 8M(2^{k-1})} \\ &= \boldsymbol{2^k C_{qr}(2^0) + \sum_{i=0}^{k} 2^{k-i} C_{qp}(2^{i-1}) + 8\sum_{i=0}^{k} 2^{k-i} M(2^{i-1})} \\ &= \boldsymbol{\gamma(\phi + 8)\sum_{i=0}^{k} 2^{k-i}(2^{\beta(i-1)})} \\ &= \boldsymbol{\gamma(\phi + 8)\frac{2^{\beta}n^{\beta} - 2n}{2^{\beta}(2^{\beta} - 2)}}\end{aligned}$$

(23)

Обмежуючись старшим ступенем за $\boldsymbol{n}$, отримуємо:

$$\begin{aligned}&\boldsymbol{C_{qr}(n) = \rho(\beta)\gamma n^{\beta},\ \rho(\beta) = (\phi + 8)/(2^{\beta} - 2)} \\ &\boldsymbol{\rho(3) = 71/48,\ \rho(\log_2 7) = 14/9}\end{aligned}$$

(24)

Алгоритм Q3RP. Описані вище властивості блоково-рекурсивних QR та QP розкладів дозволяють використовувати їх як базові операції для побудови більш складних алгоритмів матричних перетворень. Одним із них є алгоритм, який ми будемо називати Q3RP, призначений для ортогонального розкладу симетричної матриці до стрічкової форми.

Основна ідея алгоритму полягає в поетапному зануленні цілих блокових шарів симетричної матриці за допомогою послідовності QR та QP розкладів. На відміну від класичних алгоритмів, де елементи занулюються послідовними перетвореннями окремих рядків або стовпців, у цьому алгоритмі занулення виконується для цілих блоків матриці. Це дозволяє відокремлювати та незалежно обчислювати відповідні підзадачі, що забезпечує високий рівень паралелізму та робить алгоритм ефективним для обробки великих матриць на кластерах з розподіленою пам’яттю.

Нехай задано симетричну матрицю $\boldsymbol{M}$ розміру $\boldsymbol{n \times n}$. На першому етапі матриця розбивається на чотири великі блоки

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}, \tag{25}$$

де кожен блок має розмір $\boldsymbol{n/2} \times \boldsymbol{n/2}$. Далі кожен із цих блоків додатково ділиться на чотири підблоки розміру $\boldsymbol{n/4} \times \boldsymbol{n/4}$. Таким чином, матриця розбивається на 16 рівних блоків.

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix} = \begin{pmatrix} \boldsymbol{a_{1,1}} & \boldsymbol{a_{1,2}} & \boldsymbol{b_{1,1}} & \boldsymbol{b_{1,2}} \\ \boldsymbol{a_{2,1}} & \boldsymbol{a_{2,2}} & \boldsymbol{b_{2,1}} & \boldsymbol{b_{2,2}} \\ \boldsymbol{c_{1,1}} & \boldsymbol{c_{1,2}} & \boldsymbol{d_{1,1}} & \boldsymbol{d_{1,2}} \\ \boldsymbol{c_{2,1}} & \boldsymbol{c_{2,2}} & \boldsymbol{d_{2,1}} & \boldsymbol{d_{2,2}} \end{pmatrix} \tag{26}$$

Схематично це представлено на Рис. 2.

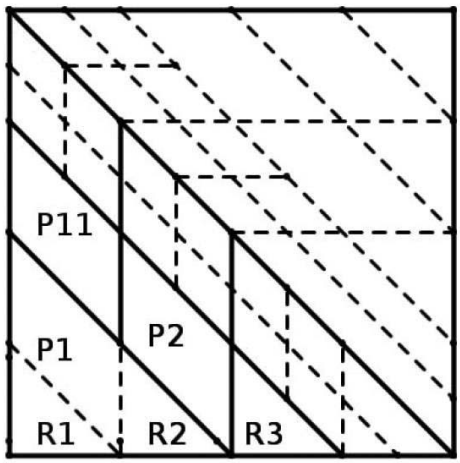

Рис. 2. Схема розбиття симетричної матриці на блоки для алгоритму Q3RP

Першим кроком алгоритму є виконання QR-розкладу блоку $\boldsymbol{c_{2,1}}$. У результаті цей блок набуває верхньотрикутної форми, а отримана ортогональна матриця $\boldsymbol{Q_1}$ задає відповідне ортогональне перетворення, яке застосовується до інших блоків матриці. Завдяки симетричності матриці таке перетворення одночасно впливає і на симетричний блок $\boldsymbol{b_{1,2}}$. Після цього виконується QP-розклад, який дозволяє усунути решту елементів у відповідному верхньотрикутному блоці. Добуток отриманих ортогональних матриць утворює узагальнену матрицю повороту $\boldsymbol{Q_3}$. Поточна матриця домножується на $\boldsymbol{Q_3^T}$ зліва та на $\boldsymbol{Q_3}$ справа, що забезпечує збереження її симетричності. На наступному етапі аналогічно обробляються блоки $\boldsymbol{c_{1,1}^U}$ та симетричний до нього блок $\left(\boldsymbol{b_{1,1}}\right)^{\boldsymbol{U}}$ за допомогою QP-розкладу. Далі послідовно застосовуються QR- та QP-розклади для блоків $\boldsymbol{c''_{2,2}}$ та $\boldsymbol{c'''_{1,2}}$. Кожен із цих кроків породжує нові ортогональні перетворення, які комбінуються у відповідну матрицю повороту для поточного шару.

У результаті послідовного застосування цих перетворень усі блоки матриці $\boldsymbol{C}$ зануляються, що приводить до повного занулення нижнього лівого блоку матриці разом із відповідними симетричними елементами верхнього правого блоку.

Завершальним етапом процедури є QR-розклад блоку $\boldsymbol{d_{2,1}^{(5)}}$. Отримана матриця повороту $\boldsymbol{Q_8}$ знову застосовується до матриці зліва та справа, що завершує формування стрічкової структури. У результаті отримуємо представлення початкової матриці у вигляді

$$\boldsymbol{S} = \boldsymbol{Q^T M Q}, \tag{27}$$

де $\boldsymbol{Q} = \boldsymbol{Q_1 Q_2 \cdots Q_8}$ є добутком ортогональних матриць, отриманих на кожному кроці алгоритму, а $\boldsymbol{S}$ — результуюча стрічкова матриця. Матриця $\boldsymbol{S}$ має спеціальну структуру: усі елементи, що знаходяться поза межами діагональної смуги шириною $\boldsymbol{p} = \frac{\boldsymbol{n}}{\boldsymbol{4}}$, дорівнюють нулю. Ширина цієї смуги визначається розміром блоків, на які розбивається початкова матриця. Отримана стрічкова форма є проміжним результатом, який надалі може бути перетворений до тридіагонального вигляду за допомогою спеціалізованих алгоритмів звуження смуги, зокрема, алгоритму Ribbon, який буде представлено в наступній статті.

Складність Q3RP. Нехай задано симетричну матрицю $\boldsymbol{M}$ розміру $\boldsymbol{2^k \times 2^k}$. Для оцінки складності алгоритму використаємо складності QR та QP розкладів, описані у попередньому розділі, та припустимо, що складність множення матриць задається функцією

$$\boldsymbol{M(n) = \gamma n^{\beta}}. \tag{28}$$

Аналіз послідовності операцій алгоритму Q3RP показує, що для матриці розміру $\boldsymbol{n}$ необхідно виконати:

- три QR-розклади для матриць розміру $\boldsymbol{n/4}$;

- три QP-розклади для матриць розміру $\boldsymbol{n/4}$;
- шість множень матриць розміру $\boldsymbol{n/2}$;
- двадцять вісім множень матриць розміру $\boldsymbol{n/4}$.

Оскільки множення матриць розміру $\boldsymbol{n/2}$ можна звести до множень блоків розміру $\boldsymbol{n/4}$, усі матричні множення можна представити як множення блоків одного розміру $\boldsymbol{n/4}$. Загальна кількість таких множень становить 50. Тож загальну складність алгоритму можна записати у вигляді

$$\begin{aligned} \boldsymbol{C_n} &= \boldsymbol{3C_{qr}(n/4) + 3C_{qp}(n/4)} \\ &\quad \boldsymbol{+ 50M(n/4)} \\ &\quad \boldsymbol{= (3\phi + 3\rho + 50)\gamma\left(n^{\beta}\right)/4^{\beta}} \end{aligned} \tag{29}$$

тобто:

$$\boldsymbol{C_n = \tau\gamma n^{\beta}}$$

$$\boldsymbol{\tau(\beta) = (3\phi + 3\rho + 50)/4^{\beta}}$$

$$\boldsymbol{\phi + \rho = \left(28\left(2^{\beta} - 1\right) + 8\left(2^{\beta}\left(2^{\beta} - 4\right)\right)\right) / \left(\left(2^{\beta} - 2\right)\left(2^{\beta}\left(2^{\beta} - 4\right)\right)\right)}$$

$$\boldsymbol{\phi + \rho = \left(8 \cdot 2^{2\beta} + 4 \cdot 2^{\beta} - 28\right) / \left(\left(2^{\beta} - 2\right)\left(2^{\beta}\left(2^{\beta} - 4\right)\right)\right)}$$

$$\boldsymbol{\tau(\beta) = \left(50 \cdot 2^{3\beta} + 412 \cdot 2^{\beta} - 84 - 276 \cdot 2^{2\beta}\right) / \left(\left(2^{\beta} - 2\right)2^{3\beta}\left(2^{\beta} - 4\right)\right)}$$

$$\boldsymbol{\tau(3) = 0.91, \quad \tau(\log_2 7) = 1.25}$$

(30)

Отже, асимптотична складність алгоритму Q3RP має той самий порядок, що і складність множення матриць. Використання швидких алгоритмів множення матриць, зокрема, алгоритму Штрассена або його узагальнень, дозволяє зменшити асимптотичний показник степеня $\boldsymbol{\beta}$ і, відповідно, покращити теоретичну оцінку складності алгоритму Q3RP. Для порівняння класичний алгоритм розкладу симетричної матриці до тридіагональної форми на основі відбиттів Хаусхолдера має складність приблизно $\frac{4}{3}\boldsymbol{n^3}$ арифметичних операцій. Однак цей метод реалізується переважно як послідовний алгоритм і не має природної блоково-рекурсивної структури, що обмежує можливості його ефективної паралельної реалізації. На відміну від нього, алгоритм Q3RP використовує блоково-рекурсивні QR та QP перетворення, що дозволяє виконувати більшість обчислень у вигляді операцій над матричними блоками. Така структура алгоритму добре узгоджується з паралельними обчислювальними середовищами та дозволяє ефективно використовувати високопродуктивні обчислювальні системи.

Експерименти

Програмна реалізація алгоритму виконана у середовищі блоково-рекурсивних паралельних обчислень DAP, яке є децентралізованою системою керування паралельними задачами [10, 11]. Система реалізована мовою Java з використанням MPI для організації взаємодії між обчислювальними вузлами. Для запуску алгоритму формується обчислювальний граф, вершинами якого є обчислювальні блоки (так звані *дропи*), а ребрами — зв'язки передачі даних між ними. Така структура дозволяє природно реалізувати рекурсивні матричні алгоритми у вигляді набору незалежних підзадач, які можуть виконуватися паралельно. У реалізації алгоритму Q3RP основні обчислювально затратні операції — QR-розклади, QP-розклади та множення матриць — додатково прискорюються за рахунок використання графічного процесора (GPU). Використання GPU дозволяє ефективно виконувати великі блокові матричні операції, що є ключовими для блоково-рекурсивної структури алгоритму. Експерименти проводилися на одній обчислювальній ноді з такими характеристиками: CPU: Intel Core i5-12600K; RAM: 32 GB DDR4; GPU: NVIDIA RTX 3070 (8 GB VRAM). Усі обчислення виконувалися у подвійній точності (double precision). Розмір листової вершини у середовищі DAP становив 128, що визначає глибину рекурсії алгоритму. Вхідні матриці генерувалися як щільні матриці зі щільністю 50 % ненульових елементів. Для кожного розміру матриці час виконання визначався як середнє значення за серією запусків.

Таблиця 1.

Час виконання та точність алгоритму Q3RP

N	Час виконання, мс	Абсолютна похибка
256	98	1.60e−15
512	121	2.34e−15
1024	354	2.71e−15
2048	1863	3.38e−15
4096	9617	4.36e−15

Отримані результати наведені в Таблиці 1 показують, що алгоритм Q3RP демонструє стабільну продуктивність при збільшенні розміру матриці.

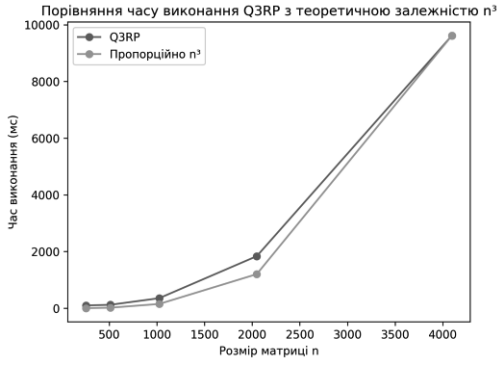

Рис. 3. Час виконання алгоритму Q3RP

З графіка на Рис. 3 видно, що зі збільшенням розміру задачі час виконання алгоритму зростає приблизно кубічно, а це відповідає теоретичній оцінці складності алгоритму.

Крім продуктивності важливою характеристикою алгоритму є його чисельна стабільність. Для оцінки точності було виміряно абсолютну похибку реконструкції матриці після виконання розкладу. Похибка визначалася як норма різниці між початковою матрицею та матрицею, відновленою після застосування відповідних ортогональних перетворень. Результати показують, що абсолютна похибка знаходиться на рівні 10^{-15}, що відповідає машинній точності для чисел типу double. Це свідчить про високу чисельну стабільність алгоритму Q3RP.

Отримані експериментальні результати підтверджують ефективність запропонованого алгоритму Q3RP для ортогонального розкладу симетричних матриць до стрічкової форми. Блоково-рекурсивна структура алгоритму дозволяє виконувати більшість обчислень у вигляді великих матричних операцій, що добре узгоджується з архітектурою сучасних обчислювальних систем.

Використання прискорення на GPU для виконання QR- та QP-розкладів і операцій множення матриць дозволяє суттєво зменшити час виконання алгоритму при зростанні розміру задачі. Водночас алгоритм зберігає високу точність обчислень, що підтверджується малими значеннями похибки реконструкції.

Висновки

У роботі описано блоково-рекурсивні алгоритми ортогонального (унітарного) розкладу матриць та отримано оцінки їхньої обчислювальної складності. Показано, що кількість арифметичних операцій визначається алгоритмом множення матриць, який використовується у відповідних блокових обчисленнях. Водночас кількість операцій у запропонованих алгоритмах у будь-якому випадку менша, ніж у двох операціях множення матриць того самого розміру. Отже, для матриць великого розміру ефективними є алгоритми, що використовують швидкі методи множення матриць.

Перевагою запропонованих алгоритмів у суперкомп’ютерних обчисленнях є їхня блокова структура. Класичні алгоритми, зокрема, алгоритм Хаусхолдера, не дозволяють природно відокремлювати незалежні блокові підзадачі, тому розпаралелювання задач великого розміру пов’язане зі значними труднощами.

Запропоновані блокові алгоритми усувають ці обмеження. Вони дають змогу виокремлювати незалежні блокові підзадачі та забезпечують паралельний і конвеєрний характер обчислювального процесу, що є особливо важливим для виконання обчислень на кластерах із розподіленою пам’яттю.

Література

1. Golub G. H., Van Loan C. F. Matrix Computations. Baltimore: Johns Hopkins University Press, 2013. 756 p.

2. Trefethen L. N., Bau D. Numerical Linear Algebra. Philadelphia: SIAM, 1997.
3. Strassen V. Gaussian elimination is not optimal. Numerische Mathematik. 1969. Vol. 13. No. 4. P. 354–356.
4. Coppersmith D., Winograd S. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation. 1990. Vol. 9. No. 3. P. 251–280.
5. Schönhage A. Unitäre Transformationen großer Matrizen. Numerische Mathematik. 1973. Vol. 20. P. 409–417.
6. Malaschonok G. Recursive matrix algorithms, distributed dynamic control, scaling, stability. Computer Science and Information Technologies (CSIT). Yerevan, 2019. P. 112–115. doi:10.1109/CSITechnol.2019.8895255.
7. Malaschonok G., Ivaskevich A. Quick recursive QR decomposition. Proceedings of the Conference on Mathematical Foundations of Informatics (MFOI-2020). Kyiv, 2020. P. 1–8.
8. Першута П. В. Алгоритм зведення симетричної матриці до тридіагональної форми із застосуванням QR та QP-розкладів: магістерська робота. Київ, 2024. 55 с.
9. Demmel J., Dumitriu I., Holtz O., Kleinberg R. Fast Linear Algebra is Stable. Numerische Mathematik. 2007. Vol. 108. P. 59–91.
10. Сідько А. А. Середовище виконання для блоково-рекурсивних матричних алгоритмів на суперкомп'ютері з розподіленою пам'яттю: дис. … д-ра філософії. Київ, 2024.
11. Малашонок Г. І., Сідько А. А. Розподілені обчислення: ДАП-технологія розпаралелювання рекурсивних алгоритмів. Наукові записки НаУКМА. 2018. № 1. С. 25–32.

References

1. Golub, G.H. and Van Loan, C.F., 2013. Matrix Computations. Baltimore: Johns Hopkins University Press.
2. Trefethen, L.N. and Bau, D., 1997. Numerical Linear Algebra. Philadelphia: SIAM.
3. Strassen, V., 1969. Gaussian elimination is not optimal. Numerische Mathematik, 13(4), pp.354–356.
4. Coppersmith, D. and Winograd, S., 1990. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation, 9(3), pp.251–280.
5. Schönhage, A., 1973. Unitäre Transformationen großer Matrizen. Numerische Mathematik, 20, pp.409–417.
6. Malaschonok, G., 2019. Recursive matrix algorithms, distributed dynamic control, scaling, stability. In: Computer Science and Information Technologies (CSIT). Yerevan, pp.112–115. https://doi.org/10.1109/CSITechnol.2019.8895255
7. Malaschonok, G. and Ivaskevich, A., 2020. Quick recursive QR decomposition. In: Proceedings of the Conference on Mathematical Foundations of Informatics (MFOI-2020). Kyiv, pp.1–8.
8. Pershuta, P.V., 2024. Algorithm for reducing a symmetric matrix to tridiagonal form using QR and QP decompositions (Master's thesis). Kyiv (in Ukrainian).
9. Demmel, J., Dumitriu, I., Holtz, O. and Kleinberg, R., 2007. Fast linear algebra is stable. Numerische Mathematik, 108, pp.59–91.
10. Sidko, A.A., 2024. Execution environment for block-recursive matrix algorithms on a distributed-memory supercomputer (PhD dissertation). Kyiv (in Ukrainian).
11. Malashonok, H.I. and Sidko, A.A., 2018. Distributed computing: DAP-technology for parallelization of recursive algorithms. Naukovi zapysky NaUKMA, 1, pp.25–32 (in Ukrainian).

Дата першого надходження до видання: 13.03.2026
Внутрішня рецензія отримана: 16.03.2026
Зовнішня рецензія отримана: 17.03.2026
Дата прийняття статті до друку: 19.03.2026
Дата публікації: 16.04.2026

Про авторів:

[1]Малашонок Геннадій Іванович,
Доктор фізико-математичних наук, професор
[1] Malaschonok Gennadiy,
Ph.D (doctor, physical and mathematical sciences), professor
https://orcid.org/0000-0002-9698-6374

[2]Сухарський Сергій Сергійович,
PhD студент
[2] Sukharskyi Sergiy,
Post-graduate student
https://orcid.org/0000-0002-5873-984X

Місце роботи авторів:

[1]Національний університет
«Києво-Могилянська академія»
[1] National University
“Kyiv–Mohyla Academy”
malaschonok@gmail.com

[2]Інститут Програмних Систем
НАН України
[2] Institute of Software Systems.
National Academy of Sciences of Ukraine
serhii.sukharskyi@gmail.com