Компьютерная математика

УДК 519.6

В.А. СИДОРУК

ГІБРИДНИЙ АЛГОРИТМ РОЗВ'ЯЗУВАННЯ ЛІНІЙНИХ СИСТЕМ З РОЗРІДЖЕНИМИ МАТРИЦЯМИ ПОПЕРЕМІННО ТРИКУТНИМ МЕТОДОМ

Розглядається новий гібридний алгоритм розв'язування систем лінійних алгебраїчних рівнянь з розрідженими симетричними додатно визначеними матрицями на комп'ютерах з графічними прискорювачами. Подано результати апробації алгоритму на багатоядерному комп'ютері з графічними прискорювачами Інпарком-G.

Вступ. Інтерес до проблеми побудови ефективних методів розв'язування СЛАР з розрідженими матрицями переважно обумовлений їх численними застосуваннями. Зокрема, системи рівнянь з розрідженими матрицями виникають у задачах аналізу міцності конструкцій в цивільному та промисловому будівництві, авіабудуванні, суднобудуванні тощо. Область застосування методів розв'язування СЛАР з розрідженими матрицями постійно розширюється.

Разом з тим вимоги до високопродуктивних обчислень набагато випереджають можливості традиційних паралельних комп'ютерів, навіть не зважаючи на багатоядерність процесорів. Розв'язання проблеми прискорення обчислень на комп'ютерах з багатоядерними процесорами розглядається в площині використання для прискорення обчислень гібридних систем на основі багатоядерних процесорів (CPU) і графічних прискорювачів GPU.

Одним із напрямків розв'язування лінійних систем з розрідженими матрицями є ітераційні процеси на основі трикутних методів [1]: метод Зейделя, метод верхньої релаксації, метод симетричної верхньої релаксації, попереміннo-трикутний метод. В роботі розглядається гібридний алгоритм попереміннo трикутного методу, в якому перевпорядкування ненульових елементів і побудова регуляризатора, виконується на CPU, а знаходження розв'язку ітераційним методом відбувається на GPU.

У даній статті розглядаються СЛАР з симетричними додатно визначеними розрідженими матрицями блочно-діагонального виду з обрамленням. Такі матриці можна отримати, застосувавши метод паралельних перерізів [2, 3].

1. Постановка задачі

Розглянемо задачу

$$Ax = b \tag{1}$$

з симетричною додатно-визначеною розрідженою матрицею порядку n.

$$A = \begin{pmatrix} A_{11} & 0 & 0 & \cdots & 0 & A_{1p} \\ 0 & A_{22} & 0 & \cdots & 0 & A_{2p} \\ 0 & 0 & A_{33} & \cdots & 0 & A_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & A_{p-1,p-1} & A_{p-1,p} \\ A_{p1} & A_{p2} & A_{p3} & \cdots & A_{p,p-1} & A_{pp} \end{pmatrix},$$

де блоки A_{ii}, A_{ip}, A_{pi} зберігають розріджену структуру.

Для розв'язання системи (1) застосуємо неявний двокроковий метод [1].

$$B\frac{x_{k+1} - x_k}{\tau_k} + Ax_k = b. \tag{2}$$

Оператор B вибирається з умови мінімізації кількості ітерацій і його економічності. До економічних методів можна віднести попеременно-трикутний метод, що описаний в [1].

Далі опишемо блочний варіант пропонованого алгоритму.

Блочний алгоритм. Розглянемо блочний алгоритм попеременно трикутного методу. Для цього матрицю $\tilde{A}$ будемо представляти як

$$\tilde{A} = R + R^T.$$

Вимоги економічності задовольняє попеременно-трикутний метод (ПТМ), суть якого полягає у побудові оператора B у спеціальному факторизованому вигляді

$$B = (E + \omega R)(E + \omega R), \tag{3}$$

де ω – ітераційний параметр, що буде визначений далі, R – оператор, якому відповідає трикутна матриця.

$$R = \begin{pmatrix} R_{11} & 0 & 0 & \cdots & 0 & 0 \\ 0 & R_{22} & 0 & \cdots & 0 & 0 \\ 0 & 0 & R_{33} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & R_{p-1,p-1} & 0 \\ R_{p1} & R_{p2} & R_{p3} & \cdots & R_{p,p-1} & R_{pp} \end{pmatrix} \tag{4}$$

Даний оператор зберігає блочну діагональну структуру $\tilde{A}$.

ПТМ належить до класу методів, ітераційні параметри для яких вибираються з урахуванням апріорної інформації про оператори ітераційної схеми. Для ПТМ ця інформація полягає у заданні величин δ та Δ, що задовольняють нерівності

$$\|x\|_A^2 \geq \delta \|x\|^2 \sqrt{b^2 - 4ac} \lim_{n\to\infty} \left\{ \sup_{\|s-s\|\gamma} |G \right\}, \ \left\| R^T x \right\|^2 \leq \Delta \|x\|_R^2 \Big/ 4 .$$

Тоді $\omega = \dfrac{2}{\sqrt{\delta\Delta}}$.

Необхідно зазначити, що в загальному випадку отримання теоретично обґрунтованих виразів для констант δ та Δ є самостійною складною задачею. Тому викликають інтерес алгоритми, які потребують мінімум апріорної інформації про задачу.

Пропонується нестаціонарна схема [1, с. 260]

$$B_{k+1} \frac{x_{k+1} - x_k}{\tau_k} + Ax_k = b. \tag{5}$$

Як і раніше B_{k+1} буде вибиратись у спеціальному факторизованому вигляді [1, с. 264]

$$B_{k+1} = \left(E + \omega_{k+1} R\right)\left(E + \omega_{k+1} R^T\right).$$

Для обчислення ω_{k+1} використаємо наступну процедуру:

$$\omega_{k+1} = \|w_k\| / \left\|R^T w_k\right\|, \ \ \omega_0 = 0.$$

Ітераційний процес будемо реалізовувати за наступними формулами:

$$r_k = Ax_k - b. \tag{6}$$

$$\left(E + \omega_k R\right)\overline{w}_k = r_k . \tag{7}$$

$$\left(E + \omega_k R^T\right) w_k = \overline{w}_k . \tag{8}$$

$$x_{k+1} = x_k - \tau_{k+1} w_k . \tag{9}$$

Вибір ітераційного параметра τ_{k+1} виконується за формулою методу скорішого спуску.

$$\tau_{k+1} = \frac{\left(w_k, r_k\right)}{\left(Aw_k, w_k\right)}. \tag{10}$$

У подальшій викладці будемо вважати, що матриця E та вектори b, x_k, x_{k+1}, w_k, $\overline{w}_k$, r_k теж мають блочну структуру, а величина i блоку відповідає порядку блоку R_{ii}. Позначення:

- $b_i, x_{k_i}, x_{k+i_i}, w_{k_i}, \overline{w_{k_i}}, r_{k_i}$, означатимуть, що ми працюємо з i частиною відповідних векторів;
- $b_p, x_{k_p}, x_{k+i_p}, w_{k_p}, \overline{w_{k_p}}, r_{k_p}$, означатимуть, що ми працюємо з частинами векторів, що відповідають останньому процесору.

Для реалізації (6) виконуємо наступні співвідношення:

$$r_{k_i} = R_{ii}x_{k_i} + R_{ii}^T x_{k_i} + R_{pi}^T x_{k_p} - b_i,\ 1 \le i < p, \tag{11}$$

$$r_{k_p} = \sum_{i=1}^{p-1} R_{pi}x_{k_i} + R_{pp}x_{kp} + R_{pp}^T x_{k_p} - b_p. \tag{12}$$

Для формули (7) реалізуються співвідношення:

$$\overline{w}_{ki} = \left(E_{ii} + \omega_k R_{ii}\right)^{-1} r_{k_i},\ 1 \le i < p, \tag{13}$$

$$\overline{w}_{kp} = \left(E_{pp} + \omega_k R_{pp}\right)^{-1}\left(r_{k_p} - \sum_{i=1}^{p-1} \omega_k R_{pi}\overline{w}_{ki}\right). \tag{14}$$

Для формули (8) діють такі співвідношення:

$$w_{kp} = \left(E_{pp} + \omega_k R_{pp}^T\right)^{-1} \overline{w}_{kp}, \tag{15}$$

$$w_{k_i} = \left(E_{ii} + \omega_k R_{ii}^T\right)^{-1}\left(\overline{w}_{ki} - \omega_k R_{pi}^T w_{kp}\right),\ 1 \le i < p. \tag{16}$$

Запишемо блочний алгоритм для (10).

Введемо два вектори z, y:

$$y = \left(w_k, r_k\right),$$
$$z = \left(Aw_k, w_k\right).$$

Елементи векторів обчислюються за наступними формулами:

$$y_i = \left(w_{ki}, r_{ki}\right),\ i = \overline{1,2,...,p}. \tag{17}$$

$$z_i = \left(\left(R_{ii}w_{ki} + R_{ii}^T w_{ki} + R_{pi}^T w_{kp}\right), w_{ki}\right),\ 1 \le i < p. \tag{18}$$

$$z_p = \left(\left(\sum_{i=1}^{p-1} R_{pi}w_{k_i} + R_{pp}w_{k_p} + R_{pp}^T w_{k_p}\right), w_{k_p}\right). \tag{19}$$

$$\tau_{k+1} = \sum_{i=1}^{p} y_i \Big/ \sum_{i=1}^{p} z_i. \tag{20}$$

Для формули (9) реалізуються співвідношення:

$$x_{k+1i} = x_{k_i} - \tau_{k+1}w_{ki},\ \ 1 \le i < p. \tag{21}$$

Паралельний гібридний алгоритм. Припустимо, що для обчислень нам доступна система архітектури p CPU + p GPU. Також будемо вважати, що пам’яті p GPU нам достатньо для збереження необхідних даних.

Реалізуємо наступну декомпозицію даних:

- у пам'яті GPU з номерами i, $1 \le i < p$ містяться блоки R_{ii}, R_{ip}, а також частини векторів r_{k_i}, x_{k_i}, x_{k+1_i}, x_{k_p}, b_i, $\overline{w}_{k_i}$, w_{k_i}, $w_{k\,p}$, y_i, z_i;
- у пам'яті GPU з номером p, містяться блок R_{pp}, а також частини векторів r_{k_p}, x_{k_p}, x_{k+1_p}, b_p, $\overline{w}_{k\,p}$, $w_{k\,p}$, y_p, z_p.

Запишемо паралельний гібридний алгоритм ПТМ.

1. У i-их GPU:
   – незалежно виконуємо (11) та знаходимо добуток $R_{pi}x_{k_i}$.
2. У p-му GPU:
   – одночасно виконуємо $r_{k_p} = R_{pp}x_{kp} + R_{pp}^T x_{k_p} - b_p$;
   – здійснюємо збирання $R_{pi}x_{k_i}$ і модифікацію $r_{k_p} = r_{k_p} + R_{pi}x_{k_i}$.
3. У i-их GPU:
   – незалежно виконуємо (13) та знаходимо добуток $\mu_k R_{pi}\overline{w}_{k\,i}$.
4. У p-му GPU:
   – здійснюємо збирання $\omega_k R_{pi}\overline{w}_{k}$ і модифікацію $r_{k_p} = r_{k_p} - \omega_k R_{pi}\overline{w}_{k}$;
   – виконуємо $\overline{w}_{kp} = \left(E_{pp} + \omega_k R_{pp}\right)^{-1} r_{k_p}$.
5. У p-му GPU:
   – виконуємо (15);
   – розсилаємо результат попереднього пункту в i-ті GPU.
6. У i-их GPU:
   – одночасно і незалежно виконуємо (16).
7. У i-их GPU:
   – незалежно виконуємо (18) та знаходимо добутки $R_{pi}w_{k_i}$.
8. У p-му GPU:
   – сумуємо знайдені добутки $R_{pi}w_{k_i}$;
   – виконуємо (19).
9. У всіх GPU виконуємо операцію (17).
10. У p-му GPU:
    – збираємо частини y_i, z_i та виконуємо (20);
    – розсилаємо результат виконання по всіх GPU.
11. У i-их GPU:
    – одночасно і незалежно виконуємо (21).
12. У p-му GPU:
    – виконуємо (21);
    – розсилаємо результат виконання по всіх GPU.

Програмна реалізація і чисельний експеримент. Розглянемо програмну реалізацію вищенаведеного паралельного гібридного алгоритму ПТМ для платформи 1 CPU + 1 GPU. Як зазначалось раніше процес перетворення матриці до блочно-діагонального вигляду проводиться на CPU, на GPU виконуємо всю арифметику.

Основні особливості реалізації алгоритму для платформи 1 CPU + 1 GPU:

1) не потрібно робити обміни даними, оскільки всі дані знаходяться в глобальній пам'яті GPU;

2) так як більшість операції в нашому алгоритмі виконується асинхронно, ми можемо їх запускати в різних потоках виконання cudaStream [4, 5]. Це дає можливість більш ефективно використати обчислювальні ресурси GPU, оскільки виконання операцій у даних потоках виконуються незалежно. На GPU можна створювати до 16 cudaStream.

Розрахунки проводились на вузлах кластеру Інпарком-G, які мають наступні характеристики:

- процесори: 2 Xeon 5606 (8 ядер) з частотою 2.13 ГГц;
- графічні прискорювачі: 2 Tesla M2090;
- обсяг оперативної пам'яті: 24 Гб;
- комунікаційне середовище: InfiniBand 40 Гбіт/с (з підтримкою GPUDirect), Gigabit Ethernet.

Чисельні експерименти проводились на розріджених матрицях, що наведені в табл. 1. Також у таблиці наведені характеристики матриці такі як: порядок матриці, кількість ненульових елементів.

ТАБЛИЦЯ 1. Набір тестових матриць з Флоридської колекції розріджених матриць

Назва	Проблемна область	Порядок	Кількість ненульових елементів
G3_circuit	circuit simulation problem	1 585 478	7 660 826
G2_circuit	circuit simulation problem	150 102	726 624
parabolic_fem	computational fluid dynamics problem	525 825	3 674 625
apache2	structural problem	715 176	4 817 870

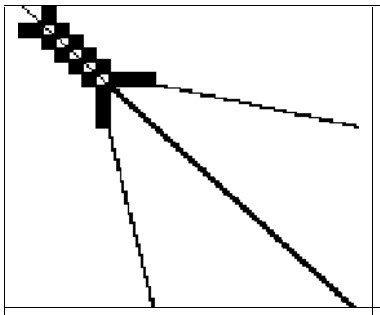
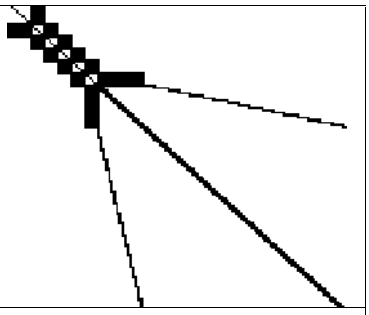
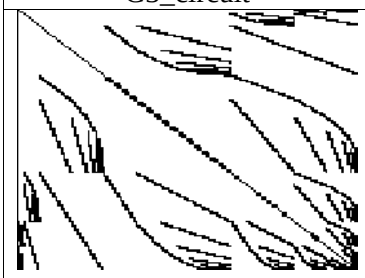
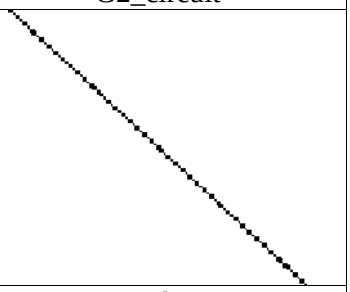
Результати розрахунків наведені в табл. 2.

ТАБЛИЦЯ 2. Співвідношення часів виконання послідовної версії алгоритму (CPU) та гібридного алгоритму (GPU) при $\varepsilon = 0.0001$ і кількості діагональних блоків рівній трьом

Назва	CPU(сек.)	GPU(сек.)	Прискорення
G3_circuit	313,58	24,757826	12,66589382
G2_circuit	73,7472	10,748288	6,861297194
parabolic_fem	207,161	21,362626	9,697356567
apache2	443,013	59,657124	7,425986504

В табл. 3 наведені портрети розріджених матриць з тестового набору.

ТАБЛИЦЯ 3. Профіль наповненості ненульовими елементами вихідних матриць

	
G3_circuit	G2_circuit
	
parabolic_fem	apache2

Висновки. Розроблено та експериментально досліджено гібридний алгоритм поперемінно трикутного методу розв'язування систем лінійних алгебраїчних рівнянь з розрідженими матрицями нерегулярної структури на комп'ютерах з графічними прискорювачами. Основою пропонованого підходу є структурна регуляризація матриць (перевпорядкування елементів матриці до блочно-діагонального вигляду з обрамленням) та застосування відомих ітераційних процедур на основі трикутних методів.

Показано, що виконання структурної регуляризації, використання на всіх етапах обчислень функцій з ефективних програмних бібліотек, а також урахування особливостей гібридної архітектури комп'ютера дають можливість отримати суттєве прискорення.

В.А. Сидорук

ГИБРИДНЫЙ АЛГОРИТМ РЕШЕНИЯ ЛИНЕЙНЫХ СИСТЕМ С РАЗРЕЖЕННЫМИ МАТРИЦАМИ ПОПЕРЕМЕННО-ТРЕУГОЛЬНЫМ МЕТОДОМ

Рассматривается новый гибридный алгоритм решения систем линейных алгебраических уравнений с разреженными симметричными положительно определенными матрицами на компьютерах с графическими ускорителями. Представлены результаты апробации алгоритма на многоядерном компьютере с графическими ускорителями Инпарком-G.

V.A. Sydoruk

HYBRID ALGORITHM FOR SOLVING LINEAR SYSTEMS WITH SPARSE MATRIX BY ALTERNATELY-TRIANGULAR METHOD

A new hybrid algorithm for solving systems of linear algebraic equations with sparse symmetric positive definite matrix on computers with GPU is considered. The results of testing of the algorithm on multicore computer Inparcom-G are presented.

1. *Самарский А.А., Николаев Е.С.* Методы решения сеточных уравнений. – М.: Наука, 1978. – 592 с.
2. *Джордж А., Лю Дж.* Численное решение больших разреженных систем уравнений. – М.: Мир, 1984. – 334 с.
3. *Писанецки С.* Технология разреженных матриц. – М.: Мир, 1988.
4. *CUDA* C Programming Guide Version 4.2. – Santa Clara: Nvidia, 2012. – 173 p.
5. *Параллельные* вычисления на CUDA. Архитектура и программная реализация модель CUDA: Учеб. пособие / А.В. Боресков, А.А. Харламов и др. Предисл.: В.А. Садовничий. – М.: Издательство Московского университета, 2012. – 336 с.
6. *Химич А.Н., Молчанов И.Н., Мова В.И. и др*. Численное программное обеспечение MIMD-компьютера Инпарком. – Киев: Наукова думка, 2007. – 222 с.

Получено 02.11.2015

Про автора:

Сидорук Володимир Антонович,
молодший науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України.