Про розв'язування систем лінійних рівнянь зі стрічковими матрицями на комп'ютерах гібридної архітектури

Хіміч О.М.[1], Баранов А.Ю.[1]

[1]*Інститут кібернетики ім. В.М. Глушкова НАН України, Київ, Україна*

khimich_ic@mail.ru, abaranov.ua@gmail.com

Анотація. *В даній статті розглядується реалізація алгоритму розв'язування систем лінійних алгебраїчних рівнянь зі стрічковими симетричними додатно визначеними матрицями на комп'ютерах з графічними прискорювачами. Подано результати апробації алгоритму на багатоядерному комп'ютері з графічними прискорювачами Інпарком[4].*

Ключові слова

Паралельні обчислення, стрічкові матриці, метод Холецького, MIMD, SIMD, CUDA.

1 Вступ

При чисельному розв'язанні науково-технічних задач в багатьох випидках виникає необхідність розв'язувати задачу (або декілька піздзадач) лінійної алгебри - систему лінійних алгебраїчних рівнянь (СЛАР). Наприклад, задачі лінійної алгебри виникають при дискретизації крайових задач чи задач на власні значення проекційно-різницевим методом (скінченних елементів, скінченних різниць). Також, при використанні ітераційних методів ровз'язання нелінійних задач часто на кожній ітерації розв'язується лінеаризована задача - СЛАР.

Важливою особливістю задач лінійної алгебри, які виникають при дискретизації, являється те, що кількість ненульових елементів матриць таких задач складає kn, де $k << n$, а n - порядок матриці, тобто матриці є розрідженими[1]. Структура розрідженої матриці визначається нумерацією невідомих задачі і часто є стрічковою, блочно-діагональною з обрамленням, профільною і тому подібне. В даній статті розглядуються симетричні додатно визначені матриці стрічкової структури.

Іншою важливою особливістю є високий порядок матриць СЛАР - до десятків мільйонів. Це зумовлюється бажанням використовувати більш точні дискретні моделі, що дає можливість отримувати наближенні розв'язки більш близькі до розв'язків вихідних задач, краще враховувати локальні особливості даного процесу чи явища.

Розв'язання таких задач потребує значних обчислювальних ресурсів, котрі забезпечуються використанням паралельних комп'ютерів. Однак, в наш час уже недостатньо продуктивності багатоядерних архітектур. Розв'язання проблеми нарощування обчислювальних ресурсів лежить в площині використання гібридних (MIMD, SIMD) архітектур. Пропонується алгоритм розв'язання СЛАР з стрічковими симетричними додатно визначеними матрицями, який використовує обчислювальні можливості CPU та GPU.

2 Алгоритм розв'язання СЛАР зі стрічковими симетричними додатно визначеними матрицями

Розглянемо систему лінійних алгебраїчних рівнянь

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

де матриця A - симетрична додатно визначена, n - порядок матриці A. Найбільш ефективним прямим методом розв'язання такої задачі є, як відомо , метод Холецького[2]. Розв'язання системи (1) полягяє в розв'язанні підзадач: трикутне розвинення матриці системи (2), розв'язання двох СЛАР з трикутними матрицями (3) та (4)

$$A = L * L^T \tag{2}$$

$$Ly = b \tag{3}$$

$$L^T x = y \tag{4}$$

Оскільки кількість арифметичних операцій при розв'язання СЛАР з трикутними матрицями (3) та (4) набагато менше (напкриклад, для стрічкових матриць в m раз, де m - ширина стрічки), в порівняння з трикутним розвиненням матриці вихідної системи, то надалі зосередимося на паралельних алгоритмах трикутного розвинення стрічкових симетричних додатно визначених матриць.

3 Декомпозиція даних

При розробці алгоритмів розв'язання задач з розрідженими матрицями особливе значення має вибір способів задання і збереження ненульових елементів. Ці способи визначаються структурою (розміщення ненульових елементів) розрідженої матриці та потребами алгоритму розв'язання задачі.

Розіб'ємо матрицю на вертикальні прямокутні плитки [3]. На рисинку 1 зображена схема розбиття стрічкової симетричної матриці. Таким чином, при реалізації блочного алгоритму Холецького з'являється можливість використовувати високоефективні фукнції cuBLAS. Іншою важливою особливостю такого задання даних є те, що на кожному кроці блочного алгоритму Холецького нам потрібно мати в пам'яті тільки m/w плиток, де m - напівширина стрічки, w - ширина плитки, які розташовані послідовно за плиткою з ведучим блоком. Таким чином, на кожному кроці потрібно копіювати в пам'ять GPU лише одну додаткову плитку. Також введемо деякі позначення, які будуть потрібні при опису алгоритму трикутного розвинення (схематично зображенно на рисунку 1):

- Pb_i - діагональний блок плитки з номером i,
- $Pu_{i,j}$ - підматриця плитки i, що починається з рядка $j * w$, де w -ширина плитки,
- $Ps_{i,j}$ - квадратна підматриця плитки i, що починається з рядка $j * w$, де w -ширина плитки.

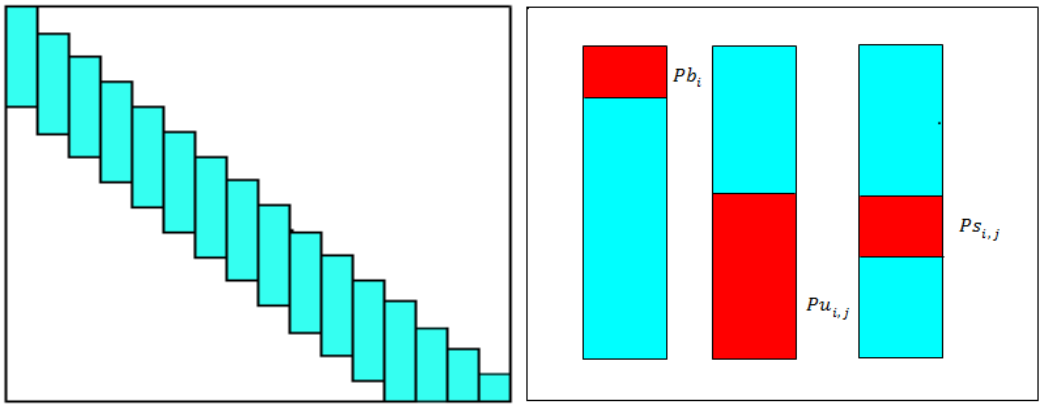

Рис. 1. Декомпозиція даних.

4 Алгоритм факторизації стрічкових симетричних додатно визначених матриць

На кожному кроці алгоритму факторизації будуть задіяні m/w плиток матриці, які розташовані підряд. Нехай номери цих плиток $i, i+1, i+2, ..., i+m/w-1$. Тоді на наступному кроці будуть задіяні плитки з номерами

$i+1, i+2, i+3, ..., i+m/w$. На початку роботи алгоритму виділимо об'єм пам'яті, якого достатньо для зберігання $m/w+1$ плитки матриці. В цій пам'яті будуть зберігатися плитки, які задіяні на даному кроці. Далі наведемо схему роботи алгоритму на i-му кроці:

- Асинхронне копіювання в пам'ять GPU плитки з номером $i + m/w$
- Факторизація діагонального блоку i-ї плитки $Pb_i \leftarrow L * L^T$, результат записується в Pb_i;
- Перетворення на GPU нижньої частини плитки $Pu_{i,1}$ за формулою $Pu_{i,1} \leftarrow Pu_{i,1} * (Pb_i)^{-1}$;
- Асинхронне копіювання в пам'ять CPU плитки з номером i, та її подальше збереження.
- Ітерації по j від 1 доки існує $Pu_{i,j}$, виконуємо на GPU перетворення $Pui+j,0 \leftarrow Pui+j,0 - Pu_{i,j} * (Ps_{i,j})^T$ (тут задіяна не вся $Pui+j,0$, а тільки перші рядки, в кількості, що дорівнює кількості рядків $Pu_{i,j}$);

Варто відзначити, що з таким підходом ми можемо факторизувати симетрічні додатно визначені стрічкові матриці довільного порядку, оскільки об'єм використаної пам'яті не залежить від порядку матриці, а є лише залежність від напівширини стрічки. Також, обміни даними між CPU та GPU проходять паралельно з обчисленнями. Після роботи алгоритму отримуємо нижню трикутну матрицю L, для якої справджується формула (2).

5 Чисельні експерименти

На основі запропонованого паралельного алгоритму була розроблена програма для комп'ютера з гібридною (MIMD, SIMD) архітектурою. Апробація програми була проведена на паралельному комп'ютері з графічними прискорювачами Інпарком.

На рисунках 2 та 3 подано залежність часу роботи програми від ширина плитки і порядку матриці, також подано порівняння з бібліотекою MKL.

6 Висновки

Для СЛАР зі стрічковими симетричними додатно визначеними матрицями запропоновано алгоритм розв'язання, який забезпечує високу ефективність розпаралелювання на GPU, враховує структуру стрічкової матриці, оптимізує використання пам'яті GPU. Також було проведено дослідження вибору оптимального розміру ширини плиток, на які розбивається вихідна матриця.

Подальші дослідження доцільно направити на розробку алгоритмів з використанням декількох CPU і декількох GPU.

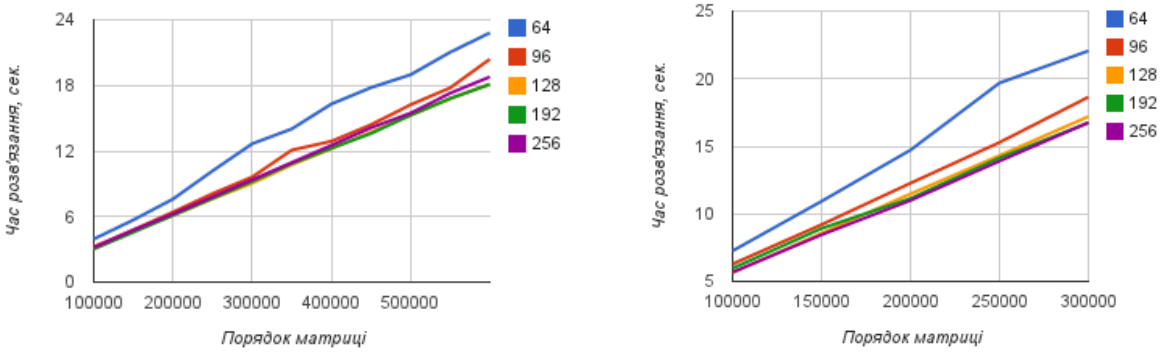

Рис. 2. Залежність часу розв'язання СЛАР від ширини плитки з різною напівшириною стрічки (1000, 2000).

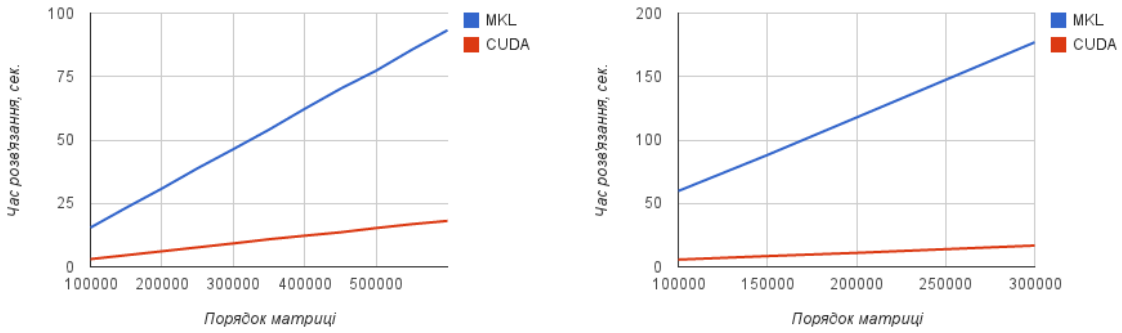

Рис. 3. Порівняння часу розв'язання СЛАР з різною напівшириною стрічки(1000, 2000) запропонованим алгоритмом з MKL.

Література

[1] А. Н. Химич, А. В. Попов, В. В. Полянко: Алгоритмы параллельных вычислений для задач линейной алгебры с матрицами нерегулярной структуры // Кибернетика и систем. анализ, - 2011. - 47, № 6. - с.159-174.

[2] Уилкинсон Дж. Х., Райнш К. Справочник алгоритмов на языке Алгол. Линейная алгебра. – М.: Машиностроение, 1976. – 389 с.

[3] Alfredo Buttari, Julien Langou, Jakub Kurzak, and Jack Dongarra: A Class of Parallel Tiled Linear Algebra Algorithms for Multicore Architectures. Parallel Computing, Volume 35, Issue 1, pp 38-53, 2009, ISSN:0167-8191

[4] Химич А.Н., Молчанов И.Н., Мова В.И. и др. Численное программное обеспечение MIMD–компьютера Инпарком. – Киев: Наукова думка, – 2007. – 222 с.