УДК 004.04

Г.В. Шимчук, Р.М. Небесний
Тернопільський національний технічний університет імені Івана Пулюя, Україна

СТРАТЕГІЯ ПАРАЛЕЛІЗМУ ДЛЯ АЛГОРИТМУ MRRR

Shymchuk, R.M. Nebesnyi

STRATEGY OF CONCURRENCY FOR MRRR ALGORITHM

Алгоритм MRRR – перший стабільний метод, який обчислює k власних пар за O(nk) арифметичних операцій [1, 2]. Завдяки своїй низькій складності, MRRR є одним з найшвидших алгоритмів для обчислення власних значень [3]. У той час як для великих завдань середовище з розподіленою пам'яттю є необхідністю, малі та середні задачі, як правило, обробляються на одному процесорі, який зараз складається з декількох обчислювальних ядер, або на системах з спільною пам'яттю. Тим не менш, сучасні реалізації для розподілених систем не можуть повністю використовувати паралелізм, який пропонується багатоядерними архітектурами. MP^3-SMP спеціально розроблена, щоб скористатися особливостями багатоядерних і багатопроцесорних систем.

При виконанні на багатоядерних архітектурах, які алгоритм будуть швидшими додатково залежить від кількості наявного паралелізму. На рисунку 1 наводимо час для обчислення всіх власних пар як функцію від кількості потоків, які використовуються; вхідна матриця розміру n = 12387 це кількість закінчених елементів моделі автомобіля. Показана стандартна імплементація DC і MRRR, також нова бібліотека MP^3-SMP. BI обчислюється близько 2-х годин і QR більше ніж 6 годин. Це значно повільніше і тому не показано.

На рисунку 1 зліва показаний час виконання, а на рисунку 1 справа – прискорення. Нахил кривої MR^3-SMP для 24 потоків позитивний, вказуючи на те, що більше апаратного паралелізму дасть ще більш високе прискорення.

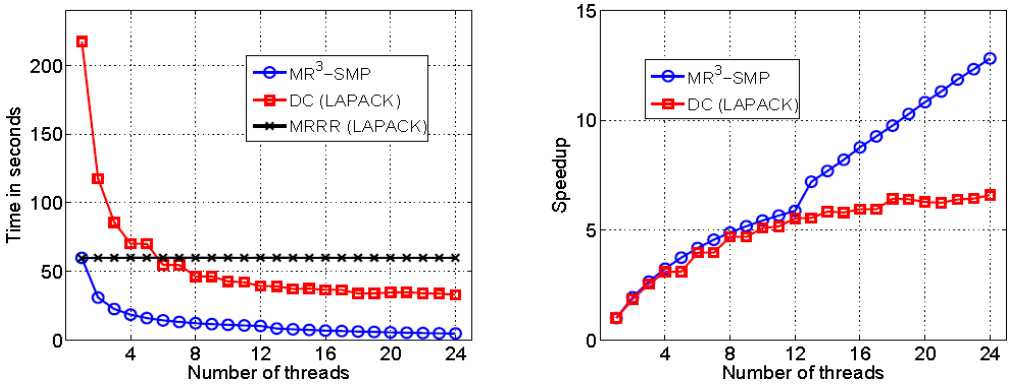

Рисунок 1. Виконання MR3-SMP та DC і MRRR для матриці розміром 12387.

Представимо конструкцію MR^3-SMP, паралельну версію алгоритму MRRR спеціально створену для багатоядерних процесорів і архітектур з спільною пам'яттю. Реалізація MR^3-SMP на C використовує потоки POSIX. Оскільки було запропоноване декілька оптимізації для підвищення точності алгоритму MRRR [3], MR^3-SMP було розроблено модульно, так щоб алгоритмічні зміни могли бути включені з мінімальними зусиллями. Версія MRRR з розподіленою пам'яттю спрямована на мінімізацію зв'язку між процесорами при досягненні оптимального балансу вимог до пам'яті [4]. Цей підхід виконується за рахунок балансування робочого навантаження, тоді як поділ праці здійснюється статично. Так як на багатоядерних архітектурах зі спільною пам'яттю про

баланс можна не турбуватись, метою є визначення правильної обчислювальної глибини, щоб краще збалансувати робоче навантаження.

Алгоритм розділяється на дві частини:

- обчислення кореневого представлення і початкового наближення власних чисел;
- розрахунок власних векторів, і остаточне уточнення власних значень.

У першій частині обчислення кореня RRR (як і для кожного підкореня RRR) займає O(n) і виконується послідовно. Початкове наближення власних значень коштує O(n) для кожного власного числа і може бути виконане паралельно з алгоритмом бісекції. Тим не менш, в залежності від доступного паралелізму, послідовний алгоритм dqds швидший, ніж бісекція і йому слід віддати перевагу [4]. Приймаючи, що бісекція використовується для обчислення власних чисел з половиною машинної точності, а алгоритм dqds сходиться протягом трьох ітерацій до власного числа, алгоритм dqds є кращим, ніж бісекція, якщо число процесорів менше або рівне 12 Г/n.

Для другої частини алгоритму, стратегія розпаралелювання полягає в розділенні обчислення на завдання, яке може бути виконане декількома потоками паралельно. Можна використовувати багато різних стратегій для поділу і планування обчислень.

Хоча власні числа уточняються у другій частині алгоритму, для простоти приймемо обчислення власного значення і обчислення власного вектора першою і другою частиною відповідно.

Оскільки дерево представлення характеризує розгортання алгоритму, природно зв'язати кожен вузол в дереві з блоком обчислень. Через таке відношення один до одного, можна взаємозаміняти поняття завдання і вузла.

В свою чергу MR^3-SMP – імплементація для знаходження власних пар, спеціально сконструйована для багатопроцесорних і багатоядерних систем. MR^3-SMP також розбиває обчислення на задачі, які виконуються паралельними потоками. Таким чином формуються три типи задач з різним пріоритетом: S-задачі, C-задачі і R-задачі відповідно. Існує декілька паралельних стратегій комбінації тих задач, але кожна гарантує уникнення будь-яких залежностей між задачами і оптимальне при певних умовах використання паралелізму. Задачі створюються і керуються динамічно. Тоді як статичний поділ роботи буде вимагати менше додаткових витрат, динамічний підхід є гнучким і здійснює чудове балансування навантаження.

Література.

1. Dhillon I. Orthogonal Eigenvectors and Relative Gaps / I. Dhillon, B. Parlett – SIAM J. Matrix Anal, 2004. – vol. 25. – C. 858-899.

2. Dhillon I. Relative Robust Representations of Symmetric Tridiagonal Linear Algebra / I. Dhillon, B. Parlett // Linear Algebra and its Applications, 2000. – vol. 309, no. 1-3. – C. 121-151.

3. Wilkinson J. The Calculation of the Eigenvectors of Codiagonal Matrices / J. H. Wilkinson. // Computer Journal, 1958. – vol. 2, no. 2. – C. 90–96.

4. Willems P. On MR3-type Algorithms for the Tridiagonal Symmetric Eigenproblem and Bidiagonal SVD / P. Willems – Disertation, University of Wuppertal, 2010