УДК 519.6

Про розв’язування систем нелінійних рівнянь на високопродуктивних обчислювальних системах з використанням прискорювачів

Нестеренко А.Н. м.н.с,

Інститут кібернетики імені В.М. Глушкова НАН України, просп. Глушкова, 40, Київ, Україна

dept150@insyg.kiev.ua

Анотація. *В роботі розглядаються деякі підходи створення алгоритмів та програм розв’язування систем нелінійних рівнянь для комп’ютерів гібридної архітектури.*

Ключові слова

Комп’ютери гібридної архітектури, технологія програмування CUDA

Моделювання будь-яких реальних процесів, що описуються системами нелінійних рівнянь (СНР) високого порядку, є великим класом задач, для розв’язування яких доцільно і необхідно використовувати високопродуктивну обчислювальну техніку. Вимоги до високопродуктивних обчислень набагато випереджають можливості традиційних паралельних комп’ютерів.

Тому для виконання великих обсягів однорідних арифметичних операцій виникає необхідність використання засобів, що прискорюють обчислення. Як прискорювачі можуть використовуватись графічні процесори (GPU). Отже постає проблема розробки паралельних алгоритмів та програмного забезпечення для гібридних систем, які поєднують можливості MIMD- і SIMD–архітектури, тобто обчислення на багатоядерних комп’ютерах з прискоренням обчислень на GPU [1].

Розглянемо питання організації обчислень при розв’язуванні СНР високого порядку на комп’ютерах гібридної архітектури (CPU+GPU). Будемо розглядати гібридну архітектуру наступного вигляду: на одне ядро приходиться одна карточка GPU Дослідження проводились на експериментальному зразку паралельного комп’ютера на графічних процесорах, який складається з двох обчислювальних вузлів, з’єднаних комунікаційною мережею. У обчислювальному вузлі використовується чотириядерний процесор. На кожному вузлі, як обчислювальні процесори використовуються дві ігрові відеокарти.

Реалізація методів розв’язування задач даних класів на комп'ютерах гібридної архітектури вимагає розподілу наступних даних в локальній пам'яті ядер кожного процесора: компонент відповідних вектор-функцій та векторів правих частин, елементів матриці Якобі, елементів вектора початкових умов, векторів наближень до розв’язку і вектора розв’язку. Крім того перераховану інформацію треба розподілити ще й між GPU (нитками графічних процесорів). Кожний процес одночасно і незалежно реалізує власну програму обчислень з використанням функцій CUBLAS на GPU [2]. Комунікаційні функції і синхронізація роботи процесів здійснюється засобами MPI [3].

Основні алгоритми розв’язування систем нелінійних рівнянь вимагають обчислення наближеного значення матриці Якобі системи: це можуть бути як різницеві формули, так і рекурентні формули обчислення наближеної матриці Якобі або оберненої до неї на послідовності ітерацій [4]. Матриця Якобі має n^2 елементів і на обчислення кожного елемента матриці потрібно витратити значну кількість операцій. В результаті на обчислення тільки матриці Якобі потрібна велика кількість операцій і відповідно значний час виконання обчислень при досить великих порядках систем.

Крім того значний час витрачається на розв’язок системи лінійних алгебраїчних рівнянь (СЛАР) або на знаходження оберненої матриці. Цей час можна суттєво скоротити, якщо при обчисленні елементів матриці Якобі і при розв’язуванні СЛАР або знаходження оберненої матриці виконання всіх арифметичних операцій розпаралелити на вибрану кількість процесів, що використовуються в комп'ютерах гібридної архітектури (CPU+GPU).

Використання гібридної архітектури розглянемо на прикладі розв'язування системи нелінійних рівнянь з оцінкою похибки розв'язку методом Деніса-Море, в обчислювальній схемі якого переважають матрично-векторні операції.

Нехай дана система n нелінійних рівнянь

$$f(x)=0, \tag{1}$$

де $f(x)=(f_1(x), f_2(x),\dots, f(x))^T$ – n-вимірна вектор-функція, а $x=(x_1, x_2,\dots, x_n)^T$ – n-вимірний вектор. Система (1) є наближенням до точної системи

$$\bar{f}(x)=0,$$

яка точно невідома, але $\|f(u)-\bar{f}(u)\|\le\Delta$. Отже після знаходження розв'язку системи (1) необхідно оцінити похибку $\|x-\bar{x}\|$.

Для розв'язування задачі (1) задаються початкове наближення $x^{(0)}$ та область, в якій шукається розв'язок, і необхідна точність ε отримання наближення до розв'язку системи. Нижнім індексом у формулах позначені номери компонент векторів, а верхнім – номери ітерацій.

Нехай $H^{(k)}=H(x^{(k)})$ – матриця Якобі системи (1) (або деяке наближення до неї). Ітераційний процес методу Деніса-Море знаходження розв'язку при заданому початковому наближенні можна записати у вигляді

$$\begin{aligned} x^{(k+1)} &= x^{(k)}-\alpha_k B^{(k)} f\left(x^{(k)}\right), \\ B^{(k+1)} &= B^{(k)}+\frac{\left(w^{(k)}-B^{(k)}y^{(k)}\right)w^{(k)T}B^{(k)}}{w^{(k)T}B^{(k)}y^{(k)}}, \end{aligned} \tag{2}$$

де $B^{(k)}=\left(H^{(k)}\right)^{(-1)}$- матриця, обернена до матриці Якобі, $w^{(k)}=x^{(k+1)}-x^{(k)}$– поправка, $y^{(k)}=f\left(x^{(k+1)}\right)-f\left(x^{(k)}\right)$, k – номер ітерації.

Зазначимо, що у ході ітераційного процесу матриця, обернена до матриці Якобі, не обчислюються, а лише уточнюється за формулою (2). Обчислення матриці Якобі і матриці оберненої до неї у ході ітераційного процесу відбувається лише при необхідності.

Паралельний алгоритм методу Деніса-Море для гібридних систем реалізується за наступною обчислюваною схемою.

1. В пам'ять кожного з p процесорних елементів MIMD-комп'ютера, тобто в пам'ять CPU, посилається вхідна інформація (порядок системи, вектор початкового наближення до розв'язку, границі області, в якій шукається розв'язок, похибка в початкових даних та ін.). Проводиться автоматичний розподіл обчислення значень компонент вектор-функції системи нелінійних рівнянь та матриці Якобі на p блоків (p - кількість процесів, що використовуються) [5].

2. Обчислюються компоненти вектор-функції $f\left(x^{(0)}\right)$ з використанням CPU. Вся вектор-функція збирається в пам'яті кожного з p процесорних елементів і посилаються в пам'ять GPU.

3. Обчислюються відповідні рядки матриці Якобі $H\left(x^{(0)}\right)$ скінченно-різницевим методом.

4. В GPU обчислюються m рядків оберненої матриці Якобі $B\left(x^{(0)}\right)$.

5. В GPU обчислюються m компонент вектора $B^{(k)} f\left(x^{(k)}\right)$.

6. Для k = 0, 1, … кожен процес:

a) за формулою $x^{(k+1)}=x^{(k)}-B^{(k)} f\left(x^{(k)}\right)$ обчислює наступне наближення до розв'язку $x^{(k+1)}$:

b) обчислює відповідні компоненти вектор-функції $f\left(x^{(k+1)}\right)$. Вся вектор-функція збирається в пам'яті CPU.

c) перевіряє умову $\left\|f\left(x^{(k+1)}\right)\right\|-\left\|f\left(x^{(k)}\right)\right\|<\varepsilon$.

– якщо умова виконана, то переходимо на пункт «d»;

– якщо умова не виконана, покладемо $x^{(0)} = x^{(k)}$, цей вектор посилається в GPU і переходимо на пункт «2»:

d) перевіряє умову закінчення ітераційного процесу

$$\left\|f\left(x^{(k+1)}\right)\right\|\le\frac{\varepsilon}{\left\|B^{(k)}\right\|}.$$

Якщо умова виконана, ітераційний процес закінчується. В протилежному випадку переходимо на пункт «e». У цьому ж пункті перевіряється умова досягнення максимальної кількості ітерацій.

e) обчислює блок відповідних рядків матриці $B^{(k+1)}$ за формулою (2)

h) В GPU обчислює m компонент вектора $B^{(k)}\, f\left(x^{(k)}\right)$.

Переходимо на пункт «a» і продовжуємо обчислення для наступного значення k.

Після завершення ітераційного процесу обчислюється похибка отриманого наближення до розв'язку задачі з наближеними даними відносно точного розв'язку системи з точними даними

$$\left\| x^{(k)} - \bar{x} \right\| \le \varepsilon + \left\| B^{(k)} \right\| \Delta ,$$

де $\bar{x}$ – точний розв'язок точної системи рівнянь.

Зауважимо, що у кожному з p процесів матрично-векторні операції виконуються з використанням функцій CUBLAS.

Нижче наведено порівняння часів розв'язку системи нелінійних рівнянь на багатоядерних комп'ютерах з використанням p процесів та на багатоядерних комп'ютерах з використанням p процесів і графічними прискорювачами.

У наведеній нижче таблиці представлені результати розв'язку системи нелінійних рівнянь

$$\begin{aligned}
&(3 - 2x_i)x_i - 2x_{i+1} + 1 = 0, && i = 0,\\
&(3 - 2x_i)x_i - 2x_{i+1} - x_{i-1} + 1 = 0, && i = 1,2,\ldots,n-2,\\
&(3 - 2x_i)x_i - x_{i-1} + 1 = 0, && i = n-1,
\end{aligned}$$

починаючи з початкового наближення $x_i = -1$ в області $D = \{a_i \le x_i \le b_i\}$, $i = 0,1,2,\ldots,n-1$, при n =4000, it = 100, eps= 1×10-10, del = 1×10-10, a_i = -1000, b_i = 1000.

Таблиця

	np=1	np=2	np=4	np=8
np CPU+np GPU	129,50	64,17	41,03	25,64
MPI		207,86	149,20	106,40
CPU	332,00			

Для порівняння також подано часи розв'язування задач без використання GPU на одному або, використовуючи паралельні алгоритми та середовище MPI, декількох (np) CPU. Із таблиці видно, що багатоядерні комп'ютери з графічними процесорами дають можливість суттєво прискорити час розв'язування задач.

Література

[1] http://www.TOP500.org/
[2] http://developer.download.nvidia.com/compute/cuda/1_0/CUBLAS_Library
[3] http://www.mpi.org/
[4] А.Н. Химич, И.Н. Молчанов, А.В. Попов, Т.В. Чистякова, М.Ф. Яковлев Параллельные алгоритмы решения задач вычислительной математики. – Киев, Наукова думка, 248 с., 2008
[5] Нестеренко А.Н., Химич А.Н., Яковлев М.Ф., Некоторые вопросы решения систем нелинейных уравнений на многопроцессорных вычислительных системах с распределенной памятью. *Вестник компьютерных и информационных технологий,* № 10:54 – 56, 2006.