О.В. Попов, канд. фіз.-мат. наук, О.В. Рудич

ДО РОЗВ'ЯЗУВАННЯ СИСТЕМ ЛІНІЙНИХ РІВНЯНЬ НА КОМП'ЮТЕРАХ ГІБРИДНОЇ АРХІТЕКТУРИ

Вступ. При чисельному моделюванні часто виникають, наприклад, при використанні тривимірних моделей, розрахункові (дискретні або напівдискретні) задачі з надвеликою кількістю рівнянь, яка може перевищувати 10^7, Причому дані (матриці) цих задач мають розріджену структуру, тобто кількість ненульових елементів значно менша (приблизно дорівнює kn, де n – порядок матриці, а $k << n$) загальної кількості елементів матриці. Зберігання таких даних, незважаючи на розрідженість, потребує значних обсягів комп'ютерної пам'яті, які можуть перевищувати 1 Tb.

Зростання параметрів задач, що розв'язуються, розрахунок на комп'ютерах більш повних моделей об'єктів, процесів, явищ вимагає відповідного зростання продуктивності комп'ютерів. Вимоги до високопродуктивних обчислень набагато випереджають можливості персональних комп'ютерів, навіть незважаючи на багатоядерність процесорів.

В даний час зростання продуктивності обчислень досягається за рахунок розпаралелювання, яке базується на використанні комп'ютерів з багатьма процесорними пристроями, зокрема з багатоядерними процесорами. В цих комп'ютерах, як правило, реалізується MIMD-архітектура (архітектура з множинним потоком команд і даних). В останні роки також набули поширення гібридні обчислювальні системи, в яких використовуються сопроцесори, наприклад, графічні процесори (GPU), для прискорення обчислень при виконанні великих обсягів однорідних арифметичних операцій. На таких сопроцесорах-прискорювачах, як правило, реалізується SIMD-архітектура паралельних обчислень. Такі комп'ютери гібридної архітектури вже зайняли провідні позиції в світовому рейтингу найпродуктивніших комп'ютерів TOP500 [1].

Розв'язування розрахункових задач в переважній більшості потребує розв'язування систем лінійних алгебраїчних задач (СЛАР) з розрідженими матрицями (див., напр., [2, 3]). В свою чергу при розв'язуванні цих СЛАР прямими методами, як правило, використовується факторизація (розвинення) матриці в добуток матриць (в більшості випадків трикутних, діагональних). Часто розвинення матриці в добуток трикутних матриць використовується і в ітераційних методах розв'язування СЛАР.

Постановка задачі. Розглянемо СЛАР

$$Ax = b, \qquad (1)$$

де A – дійсна квадратна матриця порядку n; b – матриця правої частини розміру $n \times q$ (або n-вимірний вектор). Розв'язування цієї СЛАР полягає в знаходженні такого розв'язку x (матриці розмірності $n \times q$ або n-вимірного вектора), щоб рівняння (1) перетворювалося в тотожність. Якщо матриця A системи невироджена (тобто її визначник $|A| \equiv \det(A) \neq 0$), то розв'язок СЛАР (1) існує і єдиний.

Блочні алгоритми розвинення квадратних матриць. Як зазначено вище GPU, призначені в першу чергу для виконання великих обсягів однорідних арифметичних операцій – матрично-матричних та меншою мірою матрично-векторних. До того ж такі операції ефективно реалізовано в бібліотеках програм від розробників технічних засобів, наприклад в бібліотеці CUBLAS [4]. Тому класичні методи та алгоритми (Гаусса, Холецького) доцільно модифікувати, представивши їх у блочній формі.

Розглянемо LU-розвинення. Розіб'ємо матрицю на блоки розміру $s \times s$. Не втрачаючи загальності міркувань, можна вважати, що n/s – ціле число. Після k-1 кроків ($k = 1, 2, \ldots, n/s$) блоки модифікованої матриці $A^{(k-1)}$ можна схематично представити у вигляді, який зображено на рис. 1 ліворуч. Тут позначено: $A_f^{(k-1)} = L^{(k-1)}U^{(k-1)}$ – діагональний блок (квадратний) порядку $ks–s$, $A_l^{(k-1)} = L_1^{(k-1)}U^{(k-1)}$ – піддіагональний прямокутний блок розміру $(r+s)\times(ks–s)$, $A_u^{(k-1)} = L^{(k-1)}U_1^{(k-1)}$ – наддіагональний прямокутний блок розміру $(ks–s)\times(r+s)$, $A_R^{(k-1)}$ – діагональний блок порядку $r+s$, , де $r=n–ks$. В $A_R^{(k-1)}$ в свою чергу виділяються 4 блоки: A_{11} – діагональний блок порядку s, A_{12} – прямокутний блок розміру $s \times r$, A_{12} – прямокутний блок розміру $r \times s$, A_{22} – діагональний блок порядку r.

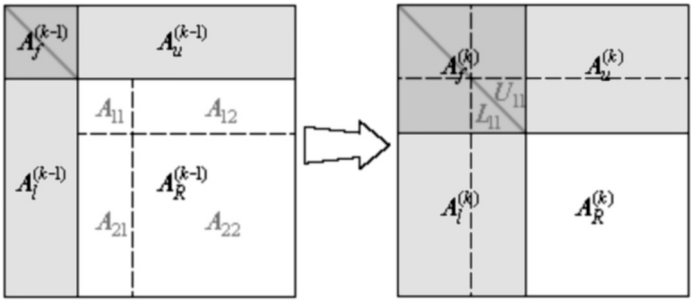

Рис. 1. Схема одного кроку блочного алгоритму LU-розвинення

На k-му кроці виконується розвинення (модифікація) блоку $A_R^{(k-1)}$ згідно формул

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = P \begin{pmatrix} L_{11}U_{11} & L_{11}U_{12} \\ L_{21}U_{11} & L_{21}U_{12} + A_R^{(k)} \end{pmatrix}, \qquad (2)$$

де P – матриця перестановок.

Спочатку згідно (2) виконується LU-розвинення блоку A_{11}. Блоки L_{21} та U_{12} згідно (2) можна отримати як розв'язки матричних СЛАР

$$U_{11}^T L_{21}^T = \widetilde{A}_{21}^T \text{ та } L_{11}U_{12} = \widetilde{A}_{12}, \qquad (3)$$

де через $\widetilde{A}_{ij}$ позначено відповідні матричні блоки після перестановок. Після цього обчислюється блок $A_R^{(k)}$ за формулою

$$A_R^{(k)} = \widetilde{A}_{22} - L_{21}U_{12}. \qquad (4)$$

Цю операцію також називають s-ранговою модифікацією.

Аналогічним чином виконується LL^T-розвинення симетричної матриці з урахуванням того, що в цьому випадку $U_{12} = (L_{21})^T$. Це дозволяє зменшити кількість арифметичних операцій майже в 2 рази. У випадку LDL^T-розвинення у формулах (2)-(4) необхідно покласти $U_{11} = D_1(L_{11})^T$, $U_{12} = D_1(L_{21})^T$, де D_1 – діагональна матриця з LDL^T-розвинення блоку A_{11}. Зауважимо, що добуток $L_{21}U_{12} \equiv L_{21}D_1(L_{21})^T$ – симетрична матриця, тому і в цьому випадку кількість арифметичних операцій зменшується майже в 2 рази, якщо обчислення проводити у такій послідовності:

(i) LDL^T-розвинення блоку A_{11},
(ii) обчислення (3) блоку U_{12} та блоку $L_{21} = (U_{12})^T(D_1)^{-1}$,
(iii) s-рангова модифікація (4).

У випадку розрідженої матриці системи A матриці розвинення L та U також залишаються розрідженими, хоча в загальному випадку кількість ненульових елементів збільшується. Тому при виконанні розвинення (2) блоку $A_R^{(k-1)}$ доцільно проводити обчислення тільки з ненульовими елементами відповідних блоків матриці. Так елемент з індексами i та j матриці (блоку) $A_R^{(k)}$ модифікується тільки в тому випадку, коли скалярний добуток i-го рядка матриці L_{21} та j-го стовпчика матриці U_{12} не дорівнює тотожно нулю. Тому s-рангова модифікація (4) виконується тільки з підматрицею матриці $L_{21}U_{12}$, яка складається з її ненульових елементів або ненульових блоків.

Структурна регуляризація розріджених матриць. Як випливає з викладеного вище, у випадку розрідженої матриці СЛАР

обробляються (модифікуються) тільки елементи вихідної матриці та всіх проміжних підматриць і блоків, які відповідають (мають ті ж самі глобальні індекси *i* та *j*) ненульовим елементам матриць розвинення. Тому кількість арифметичних операцій для розв'язування СЛАР (1) з розрідженою матрицею визначається кількістю та розташуванням ненульових елементів (тобто структурою) матриць розвинення (*L* та/або *U*) матриці системи. Отже, доцільно зберігати та обробляти тільки такі елементи, розподіляючи їх між різними процесорними пристроями відповідно до вимог конкретного паралельного алгоритму.

Структура розріджених матриць визначається нумерацією невідомих і може бути регулярною (напр., стрічковою) або нерегулярною. З метою зменшення кількості арифметичних операцій для розв'язування СЛАР (1) з розрідженою матрицею шляхом структурної регуляризації – перестановки рядків і стовпчиків (тобто перенумерації невідомих) таку матрицю приводять до одного із стандартних виглядів: стрічкового, профільного тощо. Таку процедуру варто проводити для випадків, коли немає потреби в перестановках елементів матриці з метою вибору головного елемента – для симетричних додатно визначених матриць, для матриць з діагональною перевагою тощо. Також структурна регуляризація може виконуватись при використанні блочних алгоритмів, якщо можливий частковий вибір головного елемента в межах діагонального блоку.

Існує декілька алгоритмів [2] оптимізації структури розрідженої матриці і приведення її до відповідного стандартного вигляду, наприклад:

- стрічкової або профільної матриці (в залежності від розташування ненульових елементів), використовуючи алгоритм фактор-дерев або алгоритм Катхілл-Маккі, кожен з яких забезпечує концентрацію ненульових елементів біля головної діагоналі;
- блочно-діагональної матриці з обрамленням, використовуючи алгоритм паралельних перерізів;
- матриці "хмарочосної" структури, використовуючи алгоритм мінімальної степені; така структура дозволяє найбільше (порівняно з іншими алгоритмами) зменшити кількість арифметичних операцій.

Деякі оптимізовані структури розріджених матриць представлено на рис. 2.

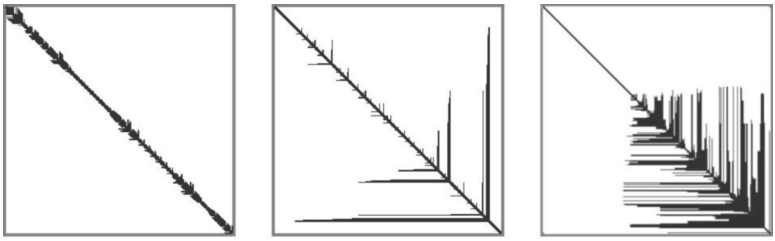

Рис. 2. Приклади структур розріджених матриць

При використанні блочних алгоритмів та GPU доцільно в блочному розбитті визначити нульові блоки (тобто блоки, які складаються тільки з нулів) та ненульові (які мають хоча б один ненульовий елемент). Цей розподіл має базуватися на вихідній структурі матриці A системи. Бажано також, щоб ненульові блоки були якомога більше заповнені. Порядок блоків s визначається, виходячи з вимог ефективного використання GPU, та коливається, як правило, в межах від 96 до 256. Практично вибір порядку блоків доцільно робити, використовуючи на конкретному гібридному комп'ютері тестові задачі аналогічні розглядуваній.

Далі необхідно оптимізувати таку блочно-розріджену структуру матриці, використавши один з названих вище алгоритмів, причому замість елементів матриці в алгоритмах використовуються блоки. Для більшого ефекту можна також оптимізувати структуру діагональних блоків, особливо якщо більшість з них мають однакову вихідну структуру. Також часто доцільно об'єднати розташовані поряд (в рядку або стовпчику) ненульові блоки оптимізованої структури матриці в один прямокутний блок.

Декомпозиція та розподіл між процесорними пристроями розріджених матриць. Всі прямі методи розв'язування СЛАР базуються на розвиненні матриці задачі в добуток матриць стандартних виглядів. Достатньо добру збалансованість завантаження процесів забезпечують паралельні версії цих алгоритмів, в яких використовуються так звані циклічні схеми розподілу і обробки матриць (див. напр. [5]). Такі циклічні алгоритми у багатьох випадках дозволяють добитися приблизно рівного об'єму обчислень і обмінів, що виконуються кожним процесом, і практично виключити вплив ефекту Гайдна [6].

Прикладом циклічної схеми розподілу між процесами елементів матриці є рядкова циклічна схема. Узагальненням даної схеми є одновимірна (рядкова) блочно-циклічна схема: якщо q-му процесу розподілено елементи рядків матриці з номерами $sr+1., sr+r,$ то $(q+1)$-му – рядки з номерами $s(r+1)+1., s(r+1)+s$, де s – кількість

рядків в блоці (можна говорити про r-й і (r+1)-й рядки квадратних матричних блоків порядку s.

У випадках стрічкових, профільних та хмарочосних матриць паралельні алгоритми, в яких використовується одновимірна блочно-циклічна схема розподілу елементів матриць, дозволяють добитися приблизно рівного об'єму обчислень і обмінів, що виконуються кожним паралельним процесом в кожний момент часу. В першу чергу – це стрічкові матриці при очевидній умові, що напівширина стрічки матриці $m > sp$. У випадку профільної матриці системи, варіюючи значення s і p, можна також практично збалансувати завантаження паралельних процесів в кожний момент часу, якщо також $c_e/n > sp$, де c_e – загальна кількість піддіагональних елементів в профілі матриці. У випадку симетричної матриці хмарочосної структури цього ж можна досягти, циклічно розподіляючи рядки блоків верхнього трикутника (з головною діагоналлю) матриці (тільки ненульові елементи або ненульові блоки).

Методологія розв'язування СЛАР. Отже, пропонується наступна послідовність дій для розв'язування СЛАР з розрідженими матрицями довільної структури на комп'ютерах гібридної архітектури:

– використовуючи один з алгоритмів структурної регуляризації, формування блочно-розрідженої структури матриці СЛАР на основі вихідної структури її ненульових елементів;

– декомпозиція розріджених матриць та розподіл отриманих рядків або стовпчиків ненульових блоків між процесорними пристроями;

– розвинення матриці СЛАР у добуток трикутних матриць – на k-му кроці ($k = 1, 2, \ldots, n/s$) виконуються операції (з можливістю асинхронного виконання на різних процесорних пристроях) з ненульовими блоками підматриці $A_R^{(k-1)}$:

(i) розвинення на CPU блоку A_{11},
(ii) обчислення (3) на GPU ненульових блоків з L_{21} та/або U_{12},
(iii) розсилка всім CPU та GPU обчислених ненульових блоків з L_{21} та/або U_{12},
(iv) s-рангова модифікація (4) на GPU ненульових блоків з A_{22};

– розв'язування на CPU двох СЛАР з трикутними матрицями $Ly = b$ та $Ux = y$ (або $L^T x = y,$ або $L^T x = D^{-1} y$).

Висновки. Запропонований підхід на основі структурної регуляризації та блочних варіантів методів трикутних розвинень матриць згідно (2)-(4) дозволяє розробити ефективні алгоритми розв'язування СЛАР з розрідженими матрицями на комп'ютерах

гібридної архітектури. Вже розроблено та програмно реалізовано алгоритми для стрічкових і профільних матриць, а також для блочно-діагональних матриць з обрамленням. При цьому досягнута висока ефективність для різних гібридних архітектур – з одним GPU, з двома GPU на одному вузлі, з g GPU на декількох вузлах. Зараз дослідження зосереджено на розв'язуванні СЛАР з матрицями довільної розрідженої структури.

Надалі доцільно розробити алгоритми дворівневої структурної регуляризації блочно-розріджених матриць – блочної структури, та структур окремих блоків.

Список використаних джерел:

1. http://www.top500.org
2. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. – М.: Мир, 1984. – 334 с.
3. Ортега Дж. Введение в параллельные и векторные методы решения линейных систем – М.: Мир, 1991. – 367с.
4. CUBLAS Linear Algebra. [Електронний ресурс], http://developer.download.nvidia.com/CUBLAS Library.pdf
5. Молчанов И.Н., Химич А.Н., Попов А.В. и др. Об эффективной реализации вычислительных алгоритмов на MIMD-компьютерах // Искусственный интеллект. – 2005, № 3. – С. 175-184.
6. Валях Е. Последовательно-параллельные вычисления. – М.: Мир, 1985. – 456 с.