УДК 519.61

БАГАТОРІВНЕВА МОДЕЛЬ ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ ДЛЯ ЗАДАЧ ЛІНІЙНОЇ АЛГЕБРИ

О.В. Попов, О.В. Рудич, О.В. Чистяков

В роботі розглядається використання багаторівневої моделі паралельних обчислень для розв'язування задач лінійної алгебри (систем лінійних алгебраїчних рівнянь та алгебраїчної проблеми власних значень) на комп'ютерах гібридної архітектури та комп'ютерах з багатоядерними процесорами Intel Xeon Phi серії x200. Сформульовано основні засади розробки алгоритмічно-програмного забезпечення таких комп'ютерів. Наведено приклади алгоритмів, які використовують багаторівневий паралелізм.
Ключові слова: багаторівнева модель паралельних обчислень, комп'ютери гібридної архітектури, процесори Intel Xeon Phi x200, системи лінійних алгебраїчних рівнянь, алгебраїчна проблема власних значень, паралельні алгоритми.

В работе рассматривается использования многоуровневой модели параллельных вычислений для решения задач линейной алгебры (систем линейных алгебраических уравнений и алгебраической проблемы собственных значений) на компьютерах гибридной архитектуры и компьютерах с многоядерными процессорами Intel Xeon Phi серии x200. Сформулированы основные принципы разработки алгоритмически-программного обеспечения таких компьютеров. Приведены примеры алгоритмов, которые используют многоуровневый параллелизм.
Ключевые слова: многоуровневая модель параллельных вычислений, компьютеры гибридной архитектуры, процессоры Intel Xeon Phi x200, системы линейных алгебраических уравнений, алгебраическая проблема собственных значений, параллельные алгоритмы.

The paper considers the use of a multilevel parallel computing model for solving linear algebra problems (systems of linear algebraic equations and an algebraic eigenvalue problem) on hybrid architecture computers and computers with multi-core processors Intel Xeon Phi x200 series. The basic principles of algorithm-software development of such computers are formulated. Examples of algorithms that use multilevel parallelism are given.
Key words: multilevel model of parallel computing, hybrid architecture computers, Intel Xeon Phi x200 processors, systems of linear algebraic equations, algebraic eigenvalue problem, parallel algorithms.

Вступ

Математичне моделювання та пов'язаний з ним комп'ютерний експеримент зараз є одним з основних засобів вивчення явищ та процесів різної природи: в суспільстві, економіці, науці, техніці тощо. Обчислювальний експеримент при створенні нових зразків енерго- і ресурсозберігаючих об'єктів істотно скорочує час і вартість створюваних об'єктів. Обчислювальний експеримент дозволяє ефективно планувати натурні експерименти і дає змогу розглядати декілька варіантів створюваних об'єктів для вибору найкращого.

Розрахункові задачі, які виникають при математичному моделюванні в галузях науки та інженерії, в багатьох випадках є задачами лінійної алгебри – системами лінійних алгебраїчних рівнянь (СЛАР) або алгебраїчною проблемою власних значень (АПВЗ). Причому, як правило, розв'язування задач лінійної алгебри займає значну частину (50 % і більше) часу розв'язання всієї задачі в цілому. Наприклад, задачі лінійної алгебри виникають при дискретизації проекційно-різницевими методами (скінчених різниць або скінчених елементів) задач аналізу міцності або стійкості конструкцій.

Важливою особливістю задач лінійної алгебри, що виникають при дискретизації, є високий порядок їх матриць (часом перевищує 10^7). Це викликано бажанням оперувати більш точними дискретними моделями, що дозволяють отримувати наближені розв'язки ближчі до розв'язків вихідних задач, краще враховувати локальні особливості процесу або явища, що розглядається. Іншою важливою особливістю є те, що матриці таких задач є розрідженими, тобто кількість їх ненульових елементів становить kn, де $k << n$, а n – порядок матриці. Розріджена структура матриці визначається нумерацією невідомих задачі і найчастіше є стрічкової, профільною, блочно-діагональною з обрамленням тощо. У багатьох випадках матриці дискретних задач симетричні і додатно визначені або напіввизначені.

Суттєве поліпшення якості математичного моделювання в багатьох галузях науки та інженерії можливе лише при використанні принципово нових тривимірних моделей, переходу від комп'ютерного моделювання окремих вузлів та агрегатів до розрахунку та оптимізації виробу в цілому. Ще одним чинником, також ресурсномістким, є забезпечення достовірності комп'ютерних розв'язків дискретних моделей, що визначається як розмірністю дискретних моделей, так і похибками вихідних даних. Очевидно, що розгляд проблем в такій постановці приводить до дискретних математичних моделей надвеликої розмірності, для комп'ютерної реалізації яких не вистачає обчислювальних ресурсів сучасних персональних комп'ютерів та робочих станцій. Такі ресурси надаються сучасними високопродуктивними комп'ютерами з паралельною організацією обчислень. В даний час суперкомп'ютери паралельної архітектури є одним з основних обчислювальних засобів моделювання складних процесів в різних прикладних областях.

Створення алгоритмів і програм з паралельною організацією обчислень потребує значного часу і дуже

високої кваліфікації користувачів. Для новітніх паралельних комп'ютерів нагальними є проблеми, пов'язані з розробкою паралельних алгоритмів та програмного забезпечення, що враховують архітектуру і технічні особливості комп'ютерів, вибір необхідної кількості та типу процесорних пристроїв, зв'язки між ними, розподіл даних задачі між цими пристроями, синхронізацію обчислень і обмінів тощо.

В цій роботі розглядається використання багаторівневої моделі паралельних обчислень для розв'язування задач лінійної алгебри (систем лінійних алгебраїчних рівнянь та алгебраїчної проблеми власних значень) на комп'ютерах гібридної архітектури та комп'ютерах з багатоядерними процесорами Intel Xeon Phi серії x200. Багаторівнева модель паралельних обчислень передбачає виділення двох основних рівнів паралелізму: верхнього – паралельно виконуються підзадачі (макрооперації), наприклад, множення матричних блоків, та нижнього – розпаралелення виконання кожної з макрооперацій. На верхньому рівні використовується MIMD-модель обчислень з розподіленою або спільною пам'яттю, а на нижньому рівні – SIMD-модель обчислень. Кожен з основних рівнів може поділятися на додаткові підрівні.

Розглянуто приклади алгоритмів, які використовують запропоновану багаторівневу модель паралельних обчислень, розв'язування двох задач лінійної алгебри: *LU*-розвинення стрічкової несиметричної матриці та часткової узагальненої проблеми власних значень для стрічкових симетричних матриць.

Високопродуктивні комп'ютери з багаторівневим паралелізмом

Впродовж декількох десятків років суттєве зростання продуктивності комп'ютерів досягається за рахунок розпаралелювання обчислень, яке базується на використанні комп'ютерів з багатьма процесорними пристроями, в тому числі різної архітектури. В перші десятиліття підвищення продуктивності досягалося за рахунок збільшення кількості процесорів (тоді одноядерних) та зростання тактової частоти цих процесорів. За останні 10-12 років відбулось суттєве зростання продуктивності обчислювальних систем з паралельною організацією обчислень – до 100 Pflops. Спершу таке зростання переважно базувалось на впровадженні багатоядерних процесорів (фірм Intel та AMD). Відзначимо, що в паралельних комп'ютерних системах з такими процесорами реалізується переважно MIMD-архітектура.

Надалі набули поширення комп'ютери гібридної архітектури, в яких багатоядерні процесори доповнюються співпроцесорами-прискорювачами. Фірми Nvidia та AMD в якості таких прискорювачів запропонували використовувати графічні процесори (GPU, відеокарти). В таких комп'ютерних системах співпроцесори-прискорювачі використовуються для виконання великих обсягів однорідних арифметичних операцій. Комп'ютери гібридної архітектури поєднують MIMD- і SIMD–архітектури. Фірма Intel запропонувала інше рішення – багатоядерні процесори Intel Xeon Phi (першого покоління) як співпроцесори.

Влітку 2016 року на ринку з'явились процесори-прискорювачі Intel Xeon Phi другого покоління (серії x200) з архітектурою Knights Landing. Процесори Intel Xeon Phi x200 з векторними арифметично-логічними пристроями (512-розрядні SIMD) здатні повністю замінити центральні процесори x86 - 64. Це, зокрема, визначає можливість виконання без перекомпіляції усіх наявних програм і зменшення складності, пов'язаної із забезпеченням одночасного використання центральних і графічних процесорів в одній системі.

Сучасні новітні зразки як графічних прискорювачів, так і процесорів Intel Xeon Phi дозволяють ущільнити високопродуктивні обчислення (в форматі персонального комп'ютера до 3,5 Tflops на подвійній розрядності). Запит на такі технології є в багатьох галузях науки та інженерії, наприклад, в задачах автоматизації проектування в галузі будівництва. Суперкомп'ютери як гібридної архітектури, так і з процесорами Xeon Phi серії x200 займають чільні позиції в світовому рейтингу найпродуктивніших комп'ютерів top500 (див. [1]).

Особливості комп'ютерів гібридної архітектури. Розглянемо коротко деякі суттєві відмінності між центральним процесором (CPU) та співпроцесором-прискорювачем (відеокарта, device, GPU).

Device – це набір мультипроцесорів GPU, на яких реалізується розпаралелювання окремих підзадач. Програма на CPU переважно виконує один потік послідовних інструкцій з максимальною продуктивністю, а графічні процесори можуть підтримувати до 1024 потоків на кожний мультипроцесор. Для розв'язування задач в технології CUDA [2] використовується дуже велика кількість ниток, що виконуються паралельно. При цьому, як правило, кожній нитці відповідає один елемент обчислюваних даних.

Програми, які реалізують розв'язування задач на CPU, можуть звертатися безпосередньо до будь-яких осередків лінійної і однорідної пам'яті. Крім того, процесори сучасних комп'ютерів мають внутрішні кеші інструкцій та даних, досить великих розмірів. Кеш-пам'ять – це швидка статична пам'ять. Вона зберігає деяку кількість останніх використаних інструкцій та даних так, що цикли і операції з масивами будуть виконуватися значно швидше. Швидкодію алгоритмів та програм на CPU можна значно підвищити, якщо розробляти їх з урахуванням наявної кеш-пам'яті процесорних ядер.

На GPU є 6 видів пам'яті, кожна з яких має своє призначення. В обчислювальних задачах здебільшого використовуються глобальна пам'ять (global memory) та розподілена пам'ять (shared memory). Глобальна пам'ять, досить велика, повільна і не кешується, використовується здебільшого для збереження необхідних даних на GPU. Розподілена пам'ять – це швидка пам'ять, яку доцільно використовувати як кеш при розв'язуванні задач. Проте ця пам'ять дуже мала у порівнянні с кеш-пам'яттю CPU. На один мультипроцесор доступно всього 16 Кбайт розподіленої пам'яті. Це необхідно враховувати при організації обчислень на GPU.

Основні проблеми паралельних обчислень для комп'ютерів гібридної архітектури пов'язані з узгодженням розподілу обчислювальних ресурсів між ядрами (CPU) процесорів та графічними прискорювачами (GPU), а також оптимізацією комунікаційних витрат між CPU та GPU.

Слід також зазначити, що алгоритми та їх програмна реалізація для паралельних комп'ютерів гібридних архітектур значною мірою залежать від технічних характеристик компонент цих комп'ютерів: пам'яті графічного процесора, пропускної спроможності мережі, співвідношення обчислювальних характеристик GPU та CPU тощо. Їх ефективна реалізація можлива лише при врахуванні таких особливостей.

Особливості комп'ютерів з багатоядерними процесорами Intel Xeon Phi. Важливою особливістю процесорів Intel Xeon Phi x200 є наявність досить великої кількості ядер (від 64 до 72), а також використання кожним з цих ядер двох VPU (векторних процесорних пристроїв), які забезпечують виконання кожним ядром тридцяти двох DP-операцій за такт. Пари ядер розташовано на плитках (tiles), а плитки з'єднано мережею топології двовимірна решітка. Така архітектура процесора створює можливість мати декілька рівнів паралелізму.

Процесори Xeon Phi мають декілька видів пам'яті. Так кожне ядро має кеш (1-го рівня) команд і даних місткістю 32 Кб, а кеш 2-го рівня місткістю 1 Мб розділяється між двома ядрами плитки. До складу процесора входять вісім модулів швидкої ("ближньої") пам'яті MCDRAM загальною місткістю 16 Гб. Пам'ять MCDRAM може працювати в трьох різних режимах: Cache Mode – як кеш 3-го рівня, який розподілено між ядрами (до 256 Мб на ядро, Flat Mode – у складі єдиного адресного простору з основною пам'яттю, Hybrid Mode – комбінований режим, коли частина MCDRAM використовується як кеш, а частина – в єдиному адресному просторі. Основна ("далека") пам'ять процесора Xeon Phi може мати місткість до 384 Гбайт, є спільною пам'яттю всіх ядер, лінійною і однорідною. Основна пам'ять складає єдиний адресний простір процесора, який може доповнюватись швидкою пам'яттю (в залежності від режиму її використання). Програми, які реалізують розв'язування задач на комп'ютері з процесором Intel Xeon Phi x200, можуть звертатися безпосередньо до будь-яких осередків основної пам'яті. Крім того існують засоби для явного розміщення даних у швидкій пам'яті. Необхідно брати до уваги, що дальня пам'ять є спільною, ближня пам'ять та кеші розподілено між ядрами (хоча і підтримується когерентність кешів). Основну пам'ять також можна розподілити між процесами.

Комп'ютери гібридної архітектури та з процесорами Intel Xeon Phi x200 існують як у форматі персональних (одновузлових) комп'ютерів, так і у форматі багатовузлових комп'ютерів (робочих станцій, суперкомп'ютерів). Так в 2011-2017 рр. Інститутом кібернетики імені. В.М. Глушкова НАН України спільно з ДП "Електронмаш" розроблено інтелектуальну робочу станцію на багатоядерних і графічних процесорах **Інпарком_G (**4 обчислювальні вузли – по 2 чотириядерні CPU Intel Xeon 5606 2.13 GHz, по 2 GPU NVIDIA Tesla M2090, 24 Gb RAM, пікова продуктивність – 5,32 Tflops на подвійній розрядності), інтелектуальний персональний суперкомп'ютер **Інпарком_pg** (2 чотириядерні CPU Intel Xeon E5520, 2 GPU NVIDIA Tesla K40, 48 Gb RAM, пікова продуктивність – від 3 Tflops на подвійній розрядності), одновузловий інтелектуальний паралельний комп'ютер **Інпарком_xp** (1 CPU Intel Xeon Phi 7210 64C, 64 ядра з 2 VPU, 16 Gb MCDRAM, 192 Gb RAM, пікова продуктивність – до 3,5 Tflops на подвійній розрядності).

Отже, архітектура новітніх багатоядерних та графічних процесорів створює умови для використання багаторівневої моделі паралельних обчислень, яка дозволяє суттєво підвищувати ефективність наявного обчислювального ресурсу.

Концепція та принципи створення алгоритмічно-програмного забезпечення

При розробці алгоритмічно-програмного забезпечення для комп'ютерів з багаторівневим паралелізмом необхідно врахувати архітектуру комп'ютера та його процесорних пристроїв, структуру їх пам'яті, зв'язки між обчислювальними пристроями тощо. Також слід передбачити можливість використання програмних засобів та інструментаріїв, які розроблено фірмами-виробниками технічних засобів (напр. Intel, NVIDIA) саме для "свого" hardware, зокрема бібліотеки Intel MKL [3], засобів NVIDIA CUDA [2].

Як зазначалось вище, багаторівнева модель паралельних обчислень передбачає виділення двох основних рівнів паралелізму: верхнього – паралельно виконуються підзадачі (мікрооперації) та нижнього – розпаралелення виконання кожної з макрооперацій.

На верхньому рівні використовується MIMD-модель обчислень з розподіленою або спільною пам'яттю. Тут, як правило, використовують частину ядер комп'ютера для паралельної реалізації макрооперацій. В залежності від моделі пам'яті розпаралелювання між процесами або потоками доцільно здійснювати засобами MPI [4] або OpenMP [5] відповідно.

На нижньому рівні використовується SIMD-модель обчислень. Тут кожна з макрооперацій верхнього рівня розпаралелюється між деякою кількістю потоків (ниток) на вільних ядрах та/або графічних прискорювачах. Для розпаралелювання на цьому рівні між нитками на ядрах доцільно використовувати

програмні модулі математичної бібліотеки Intel MKL (переважно її складової частини – бібліотеки BLAS), а на кожному GPU – засоби NVIDIA CUDA.

Ще один паралелізм може виникати за рахунок використання векторних процесорних пристроїв (VPU). Цей рівень реалізується автоматично самими технічними засобами. Але при розробці алгоритмів та програм для комп'ютерів з такими процесорами (напр., Intel Xeon Phi x200) необхідно передбачити умови ефективного використання цього паралелізму.

Обсяг обчислень для кожного обчислювального елементу, що використовується, повинен бути приблизно однаковий – це дозволить забезпечити рівномірне обчислювальне завантаження (балансування) обчислювальних елементів. Крім того, також зрозуміло, що розподіл підзадач між обчислювальними елементами має бути виконано таким чином, щоб кількість інформаційних зв'язків (комунікаційних взаємодій) між підзадачами була мінімальною.

Врахування особливостей зв'язків між процесорними пристроями. При реалізації підзадач на багатоядерних процесорах необхідно враховувати особливості зв'язків між процесорами та між ядрами в середині процесорів. Є декілька рівнів таких зв'язків: між обчислювальними вузлами, між процесорами одного вузла, між ядрами одного процесора.

Використання розподіленої пам'яті створює певні проблеми щодо обмінів даними між процесами – операцій, які залежать від реалізації (встановленої на комп'ютері) стандарту MPI та за тривалістю можуть значно перевершувати арифметичні операції та операції звернення до пам'яті. Отже, необхідно передбачати таке розміщення даних в пам'яті процесорів, при якому співвідношення пересилок даних та арифметичних операцій, що одночасно виконуються, буде збалансованим і мінімізуватиме загальний час виконання програми. З іншого боку середовище MPI створює досить сприятливі умови для асинхронного виконання обмінів даними (на фоні виконання арифметичних операцій). Середовище MPI надає можливість створювати віртуальні топології (тобто топологію зв'язків між процесами, що створюється програмно: кільце, решітка, гіперкуб тощо) з визначеної кількості процесів (ядер). Тоді кінцевому користувачу для ефективної роботи паралельної програми треба визначити необхідну кількість процесів для розв'язування задачі та розподілити вихідні дані між ними, забезпечуючи рівномірне їх завантаження та мінімізацію обмінів.

Використання спільної пам'яті та середовища OpenMP знімає проблеми обмінів даними між потоками, але створює проблеми коректного використання даних, потребуючи додаткових синхронізацій та обмінів даними між основною пам'яттю та кешами різних рівнів. До того ж середовище OpenMP краще використовувати для синхронного виконання паралельними потоками однакових підзадач з різними даними, а реалізація виконання різних макрооперацій (до того ж асинхронного) створює значні труднощі. В цьому середовищі для ефективної реалізації паралельних обчислень також необхідне рівномірне (збалансоване) завантаження потоків, що використовуються.

Отже, якщо кількість інформаційних зв'язків (комунікаційних взаємодій) між підзадачами досить велика (порівняно з виконанням арифметичних операцій) або паралельно мають виконуватися однакові підзадачі з різними даними, то доцільно реалізовувати відповідні алгоритми в середовищі OpenMP. Якщо ж в алгоритмі мінімізовано кількість інформаційних зв'язків і виконуються переважно арифметичні операції, то можливе використання середовища MPI.

Основні проблеми паралельних обчислень для комп'ютерів гібридної архітектури пов'язані з узгодженням розподілу обчислювальних ресурсів між CPU (ядрами процесорів) та GPU (графічними прискорювачами), а також оптимізацією комунікаційних витрат між CPU та GPU. Для підвищення ефективності гібридних алгоритмів необхідно використовувати асинхронне виконання операцій (виконання програм на GPU, копіювання даних між CPU і GPU та всередині GPU), а також декілька GPU з одним CPU (застосовуючи, наприклад, механізм потоків cudaStream).

Бібліотека Intel MKL є потокобезпечна, підтримує розпаралелювання і оптимізована під багатоядерні системи. Якщо доцільно використовувати багатопоточність, за допомогою Intel MKL розробник може підвищити продуктивність програм. Головною перевагою нових версій цієї бібліотеки є якнайповніше врахування можливостей та особливостей процесорів Intel, через те, що розробник технічних та програмних засобів є фірма Intel.

Одна з особливостей Intel MKL – незалежність від компілятора. Це означає, що програмний код, написаний одного разу для однієї системи, вільно переноситься на інші системи. Крім того, компоненти бібліотеки розподілені по незалежним рівням. Передбачено як розпаралелювання для компіляторів Intel, так і для компіляторів GNU.

Ще одним достоїнством бібліотеки Intel MKL є її конкурентоспроможна продуктивність на процесорах інших виробників, крім Intel, що спрощує використання бібліотеки в масових продуктах.

Засоби NVIDIA CUDA [2]. Використання нових версій засобів NVIDIA CUDA дає можливість спростити паралельне програмування та дозволити широкому колу розробників задіяти GPU в своїх програмах. Це нове програмне середовище створює наступні основні можливості:

– технологія NVIDIA GPUDirect забезпечує рівноправний зв'язок між GPU, що спрощує і прискорює

мультипроцесорне програмування і роботу прикладних програм;

– уніфікована віртуальна адресація (UVA) організовує єдиний адресний простір для основної пам'яті процесора (вузла) і пам'яті GPU, що дещо спрощує паралельне програмування та прискорює виконання програми.

Технологія NVIDIA CUDA включає й інші функції і можливості, наприклад, MPI інтеграція з прикладними програмами CUDA-модифікація MPI, така як OpenMPI, автоматично пересилає дані з та в пам'ять GPU по Infiniband, коли програма посилає або отримує запит з MPI. Крім того, доцільно використовувати готові ефективні інструменти та бібліотеки з розпаралленням обчислень на CPU та GPU, в тому числі оптимізовані бібліотеки в складі CUDA Toolkit: CUBLAS, CUFFT, CUSPARSE, MKL тощо.

Принципи дослідження та розв'язування задач лінійної алгебри. З метою більшої ефективності використання можливостей, які надає багаторівнева модель паралельних обчислень, доцільно модифікувати наявні алгоритми (див., напр., [6-8]) розв'язування задач лінійної алгебри – систем лінійних алгебраїчних рівнянь (СЛАР) та алгебраїчної проблеми власних значень (АПВЗ). Одним з основних напрямків такої модифікації є розробка блочних версій відповідних алгоритмів, яка передбачає виділення підзадач з великими обсягами однорідних арифметичних операцій, наприклад, матрично-матричних, для реалізації яких можна використати програмні модулі відповідних бібліотек, наприклад, Intel MKL, CUBLAS тощо.

Матриці багатьох розрахункових задач, які виникають при математичному моделюванні, в більшості випадків є розрідженими. Тобто кількість ненульових елементів таких матриць дорівнює kn, де $k << n$, n – порядок (кількість рядків) матриці. При цьому ненульові елементи можуть розташовуватися некомпактно. Обробка матриць розрідженої структури, створює певні труднощі при розробці ефективних паралельних алгоритмів. Ще одним факторам, який впливає, зокрема, на ефективність реалізації однієї з найбільш ресурсоємних операцій з матрицями – розвинення вихідної матриці, є зростання щільності розрідженої матриці при розвиненні. Кількість арифметичних операцій при розвиненні розрідженої матриці (а також при інших операціях з матрицями розвинення) визначається кількістю та структурою ненульових елементів матриць розвинення.

Для зменшення кількості арифметичних операцій при розвиненні розріджених матриць проводиться оптимізація (структурна регуляризація) розрідженої структури матриць розвинення. Існує низка алгоритмів переупорядкування розріджених матриць (Катхілл-Маккі, фактор-дерев, паралельних перерізів, мінімальної степені тощо), в яких перестановками рядків і стовпчиків матриці оптимізується (за певним критерієм) структура матриці. Такі переупорядкування дозволяють зменшити загальну кількість арифметичних операцій при розвиненні, але отримувані структури матриць більш ефективні для реалізації матрично-векторних операцій, ніж матрично-матричних. Тому для ефективної реалізації останніх операцій доцільно спочатку так переупорядкувати (або сформувати) розріджену матрицю, щоб переважна більшість ненульових елементів була розташована у майже щільних матричних блоках, розмір яких найбільш оптимальний для процесорних пристроїв та програмних модулів бібліотек, які використовуються.

Отже, при створені паралельних алгоритмів розв'язування задач лінійної алгебри на комп'ютерах з багаторівневим паралелізмом необхідно вирішити наступні завдання:

– визначити для яких видів матриць ефективно використовувати комп'ютери такої архітектури;

– провести структурну регуляризацію матриць задачі з метою ефективного використання багаторівневої моделі паралельних обчислень;

– розподілити задачі на підзадачі та виявити інформаційні залежності між ними з метою ефективного розпаралелення, використовуючи різні рівні паралелізму, та вибрати середовище розпаралелювання;

– врахувати структуру пам'яті процесорних пристроїв для забезпечення високої швидкодії алгоритму;

– розподілити дані та обчислення між процесорними пристроями, щоб забезпечити рівномірне завантаження (балансування) обчислювальних елементів комп'ютера.

Постановки та методи розв'язування задач лінійної алгебри

Системи лінійних алгебраїчних рівнянь (СЛАР). Розглянемо сумісну систему

$$Ax = b, \quad A \in M^{n\times n}, \quad b, x \in M^{n\times q}, \tag{1}$$

де $M^{n\times q}$ – множина прямокутних (в загальному випадку) матриць розміру $n\times q$. Розв'язок СЛАР (1) існує і єдиний, якщо матриця A невироджена (тобто її визначник $\left| A \right| \equiv \det(A) \neq 0$).

Алгебраїчна проблема власних значень (АПВЗ, алгебраїчна задача на власні значення) полягає в

знаходженні таких чисел λ, при яких існують відмінні від нульового розв'язки системи лінійних алгебраїчних рівнянь

$$A x = \lambda B x, \quad A, B \in M^{n \times n}, \quad x \in C^n, \quad \lambda \in C, \tag{2}$$

де $M^{n \times n}$ – множина квадратних матриць порядку n. Числа λ називають власними значеннями задачі (2), а вектори x – власними векторами цієї задачі. Задача (2) називається узагальненою проблемою власних значень. Якщо B – одинична матриця n-го порядку (тобто $B \equiv I_n$), то задачу

$$A x = \lambda x \tag{3}$$

називають стандартною проблемою власних значень. В цьому випадку числа λ і вектори x називають також власними значеннями і векторами матриці A.

Може бути поставлено такі задачі на власні значення:

– повна проблема власних значень – знайти всі власні значення та всі власні вектори;

– часткова проблема власних значень – знайти одне або декілька власних значень і відповідних їм векторів або тільки власні значення (всі або декілька).

При розв'язуванні прикладних задач рідко виникають СЛАР або АПВЗ з точними вихідним даними:

$$A^* x^* = b^*, \quad \text{або} \quad A^* x^* = \lambda^* B^* x^*, \quad \text{або} \quad A^* x^* = \lambda^* x^*. \tag{4}$$

Найтиповішою є постановка задач (1)–(3) і завдання відповідних похибок у вихідних даних:

$$\| A - A^* \| = \| \Delta A \| \le \varepsilon_A \| A \|, \quad \| b - b^* \| = \| \Delta b \| \le \varepsilon_b \| b \|, \quad \| B - B^* \| = \| \Delta B \| \le \varepsilon_B \| B \|. \tag{5}$$

При цьому припускається, що структура матриць (матриці) вихідної задачі (4) і збуреної задачі (1), (5), або (2), (5), або (3), (5) не змінюється.

Розв'язування всієї задачі з наближеними вихідним даними полягає в дослідженні математичних властивостей задачі (1), (5), або (2), (5), або (3), (5) , у визначенні одного з допустимих розв'язків цієї задачі та оцінці спадкових і обчислювальних похибок розв'язків.

Розв'язування СЛАР з дійсними матрицями можуть проводитися прямими та ітераційними методами. Більшість **прямих методів** заснована на ідеї послідовних еквівалентних перетворень заданої системи з метою виключення невідомих з частини рівнянь. В результаті система (1) перетвориться в еквівалентну їй систему:

$$A^{(k)} x = b^{(k)} \tag{6}$$

з матрицею $A^{(k)}$ більш простої форми, наприклад, трикутної або діагональної, тобто такої, що розв'язування еквівалентної системи (6) вже не складає труднощів.

Процес еквівалентних перетворень можна представити як послідовні множення матриці A і правої частини b зліва на матриці M_i (i = 1, 2, …, k), причому в результаті одного множення анулюється принаймні один елемент перетворюваної матриці. Матриці M_i можуть бути трикутними, ортогональними. Різні модифікації методів виключення є по суті методами розвинення матриці A в добуток трикутних матриць, ортогональної і трикутної матриць тощо.

В такому разі в задачі розв'язування СЛАР, як правило, можна виділити три підзадачі: (i) розвинення матриці системи $A = LR$, (ii) розв'язування СЛАР з лівою матрицею (пряма підстановка або прямий хід) $Ly = b$, (iii) розв'язування СЛАР з правою матрицею (зворотна підстановка або зворотний хід) $Rx = y$.

Для розв'язування СЛАР з невиродженими матрицями використовуються класичні методи – Гаусса, Холецького тощо [6, 7]. Для розв'язування СЛАР з симетричною додатно напіввизначеною матрицею можна використати (особливо у випадку розрідженої матриці системи) метод триетапної регуляризації [7], який дозволяє обчислити наближення до нормального узагальненого розв'язку.

Названі вище методи забезпечують розв'язування задач з найменшими витратами часу і пам'яті комп'ютера в порівнянні з усіма іншими прямими методами, що використовують розвинення вихідної матриці на множники. В деяких випадках для розв'язування СЛАР (як правило, з розрідженою матрицею) доцільно використати ітераційні методи.

Методи розв'язування АПВЗ умовно можна розділити на дві групи:

– методи, що використовують подібні перетворення матриць;

– методи, що базуються на властивостях підпросторів Крилова із застосуванням оптимізуючої процедури Релея-Рітца.

Для розв'язування часткових АПВЗ (2) або (3) використовуються методи, що базуються на властивостях підпросторів Крилова. Якщо знаходяться мінімальні власні значення і відповідні їм власні вектори, то для розв'язання задачі можна використовувати **метод ітерацій на підпросторі** [6, 8]. Цей метод є узагальненням методу обернених ітерацій і полягає в побудові послідовності підпросторів E_t ($t = 1, 2, \ldots$), яка збігається до підпростору E_∞, що містить шукані власні вектори. Якщо в (2) матриця $B \equiv D$ є діагональною додатно визначеною, то узагальнена АПВЗ легко зводиться то стандартної АПВЗ (3) вигляду $Cy = \lambda y$, де $C = D^{-1/2} A D^{-1/2}$, $y = D^{1/2}x$. Після цього для розв'язування названих вище часткових АПВЗ можна використати **метод бісекції** для знаходження необхідних власних значень та **метод обернених ітерацій** (оберненої степені) для обчислення власних векторів, які відповідають знайденим власним значенням.

Алгоритми розв'язування задач лінійної алгебри

Розв'язування СЛАР займає ключове місце серед багатьох задач лінійної алгебри. Ця задача часто є складовою частиною більш складних задач, наприклад, розв'язування СЛАР з напіввизначеною матрицею, часткової узагальненої алгебраїчної проблеми власних значень тощо. При розв'язуванні лінійних систем найбільше арифметичних операцій припадає на факторизацію (розвинення в добуток матриць) матриці задачі. Тому далі розглянемо алгоритми розвинення матриць. Як зазначалось вище багаторівнева модель паралельних обчислень передбачає використання блочних версій алгоритмів класичних методів розвинення.

Блочний алгоритми розвинення квадратних невироджених матриць. Розглянемо *LU*-розвинення квадратної матриці A порядку n. Розіб'ємо її на блоки розміру $s \times s$. Не втрачаючи загальності міркувань, можна вважати, що n/s – ціле число. Після $k-1$ ($k = 1, 2, \ldots, n/s-1$) кроків блоки модифікованої матриці $A^{(k-1)}$ можна схематично представити у вигляді, який зображено на рис. 1 ліворуч. Тут позначено – блоки, для яких отримано *LU*-розвинення: $A_f^{(k-1)} = L_1^{(k-1)} U_1^{(k-1)}$ – квадратний діагональний блок порядку $ks - s$, $A_l^{(k-1)} = L_2^{(k-1)} U_1^{(k-1)}$ – піддіагональний прямокутний блок розміру $(r+s)\times(ks-s)$, де $r=n-ks$, $A_u^{(k-1)} = L_1^{(k-1)} U_2^{(k-1)}$ – наддіагональний прямокутний блок розміру $(ks-s)\times(r+s)$, та квадратний діагональний блок порядку $r+s$ $A_R^{(k-1)} = P^{(k-1)} A_R^{(0)} - L_2^{(k-1)} U_2^{(k-1)}$, розвинення якого ще продовжується.

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

$$A_R^{(k-1)} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = P_k \begin{pmatrix} L_{11}U_{11} & L_{11}U_{12} \\ L_{21}U_{11} & L_{21}U_{12} + A_R^{(k)} \end{pmatrix}, \tag{7}$$

де A_{11} – квадратний діагональний блок порядку s (провідний блок k-го кроку), A_{12} – прямокутний блок розміру $s \times r$, A_{21} – прямокутний блок розміру $r \times s$, A_{22} – квадратний діагональний блок порядку r), P_k – матриця перестановок k -го кроку.

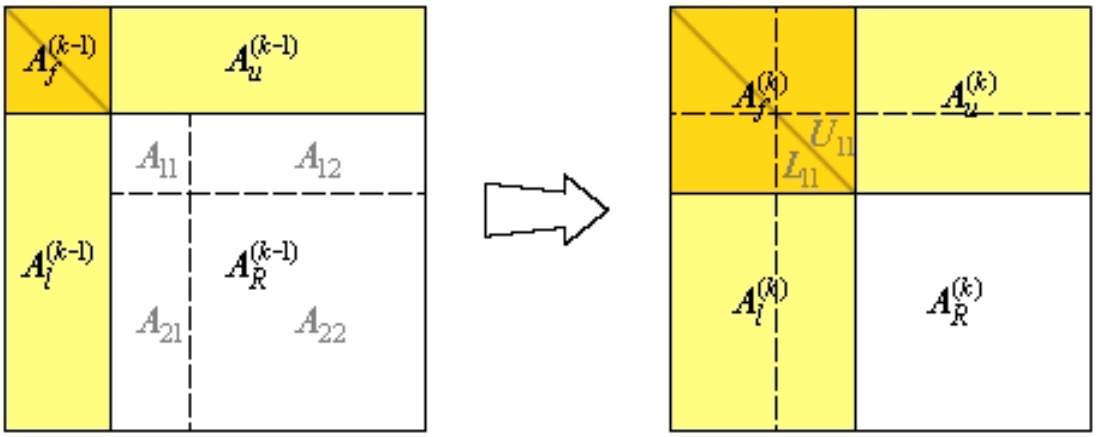

Рис. 1. Схема одного кроку блочного варіанта *LU*-розвинення

Спочатку згідно (7) методом Гаусса виконується *LU*-розвинення блоку A_{11} з вибором головного елемента. Блоки L_{21} та U_{12} згідно (7) можна отримати як розв'язки матричних СЛАР $U_{11}^T L_{21}^T = \widetilde{A}_{21}^T$ та $L_{11}U_{12} = \widetilde{A}_{12}$, де через $\widetilde{A}_{ij}$ позначено відповідні матричні блоки після перестановок. Але якщо виконується пошук головного елемента, наприклад, у стовпчику, то має проводиться одночасне розвинення блоків A_{11} та A_{21}. Блок $A_R^{(k)}$ обчислюється за формулою (цю операцію також називають *s*-ранговою модифікацією)

$$A_R^{(k)} = \widetilde{A}_{22} - L_{21}U_{12}\,. \tag{8}$$

У випадку розрідженої матриці системи A матриці розвинення L та U також залишаються розрідженими, хоча в загальному випадку кількість ненульових елементів збільшується. 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 (8) виконується тільки з підматрицею матриці $L_{21}U_{12}$, яка складається з її ненульових елементів або ненульових блоків.

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 (7), (8) 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.

Паралельний алгоритми розвинення квадратних невироджених матриць. Використовуючи блочний алгоритм, для багаторівневої моделі паралельних обчислень доцільно модифікувати алгоритм розвинення стрічкових матриць з [6] (для комп'ютерів гібридної архітектури запропоновано алгоритми такого типу в [7, 8]).

Матрицю СЛАР поділено на квадратні блоки порядку s (для спрощення викладу вважатимемо, що $n = Ns$, $m_l = M_L s$, $m_l + m_u = M_U s$, де m_l та m_u – кількість піддіагоналей та наддіагоналей відповідно). Тоді для кожного $I = 1, \ldots, N$ необхідно виконати наступні макрооперації (розв'язати підзадачі), використовуючи p процесів або потоків вищого рівня паралелізму:

1) *LU*-розвинення (з частковим вибором головного елемента) прямокутної підматриці, яка складається з блоків $A_{I,I}$, $A_{I+1,I}$, …, $A_{K,I}$, де $K = \min(I+M_L, N)$;

2) перестановка рядків в тих підмарицях, де це необхідно, використовуючи матрицю P_I;

3) обчислення рядка блоків $U_{I,I+1}$, $U_{I,I+2}$, …, $U_{I,J}$ матриці U, де $J = \min(I+M_U, N)$, як розв'язок

матричної СЛАР з нижньою трикутною матрицею;

4) для $I < N$ s-рангова модифікація (8) блоків $A_{M,L}$, де $M = I+1, \ldots, K$ та $L = I+1, \ldots, J$.

Зважаючи на викладений вище розподіл одного кроку алгоритму на макрооперації, доцільно між процесами (потоками) вищого рівня паралелізму циклічно розподіляти стовпчики блоків матриць. Наприклад, блоки, які знаходяться в стовпчику з номером t, розподіляються потоку з логічним номером $(t-1) \bmod p$. В такому разі ефективність алгоритму можна підвищити, якщо виконувати розвинення (п. 1) одночасно (паралельно) із закінченням виконання макрооперацій (пп. 3, 4) попереднього кроку, використовуючи різні процеси (потоки) верхнього рівня паралелізму.

Аналіз макрооперацій пп. 1–4 показав, що більшість обчислень можна реалізувати на нижчих рівнях паралелізму, використовуючи програмні модулі для матрично-матричних (або матрично-векторних) операцій від розробників технічних засобів.

Зауважимо, що на I-му кроці цього алгоритму операції проводяться тільки з блоками підматриці розміру $(J+1)s \times (K+1)s$, лівий верхній блок якої $A_{I,I}$. Кількість арифметичних операцій можна зменшити, якщо виключити операції з останніми нульовими стовпчиками прямокутного блоку U_{12}. В цьому разі на I-му кроці алгоритму операції проводитимуться з підматрицею розміру $(j_I - Is+s) \times (K+1)s$, де j_I – максимальне значення другого індексу ненульових елементів блоку U_{12}.

Слід зазначити, що при реалізації макрооперацій типу пп. 3, 4 (на нижчих рівнях паралелізму) доцільно об'єднувати квадратні блоки відповідного стовпчика блоків в прямокутні блоки, розмір яких має бути оптимальним з точки зору кешування даних (щоб оптимізувати обміни даними між пам'ятями різної швидкодії). З цих же міркувань має вибиратися і схема зберігання (стовпчикова чи рядкова) елементів цих блоків.

Паралельний алгоритм методу ітерацій на підпросторі [9] розв'язування часткової АПВЗ з розрідженими симетричними матрицями на комп'ютерах гібридної архітектури, який використовує, представлено в [10]. Цей же алгоритм без суттєвих змін можна використовувати і на комп'ютерах з багатоядерними процесорами Intel Xeon Phi x200, розуміючи під CPU (для розпаралелювання на верхньому рівні) частину процесорних ядер, а для розпаралелювання на нижньому рівні замість GPU використати вільні процесорні ядра.

Використання багаторівневої моделі паралельних обчислень дозволяє в багатьох випадках підвищити ефективність алгоритму за рахунок ущільнення обчислень, збільшуючи розмірність підпростору за наявності вільного обчислювального ресурсу. Адже ітераційний процес збігається лінійно, причому швидкість збіжності визначається відношенням $\lambda_q \lambda_1$, де q − розмірність підпростору, а ефективність розпаралелення на нижньому рівні тим вища, чим повніше використовується наявний обчислювальний ресурс.

Висновки

Використання багаторівневої моделі паралельних обчислень дозволяє за рахунок ущільнення обчислень (на основі структурної регуляризації та блочних варіантів методів трикутних розвинень матриць) розробити ефективні алгоритми розв'язування задач лінійної алгебри (систем лінійних алгебраїчних рівнянь та алгебраїчної проблеми власних значень), в тому числі з розрідженими матрицями, на комп'ютерах гібридної архітектури та комп'ютерах з багатоядерними процесорами Intel Xeon Phi серії x200. 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 архітектур комп'ютерів. Зараз дослідження зосереджено на розв'язуванні задач лінійної алгебри з матрицями довільної розрідженої структури.

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

Література

1. TOP 500 – 2017(11). http://www.TOP500.org/
2. CUDA TOOLKIT 4.0. http://developer.nvidia.com/cuda-toolkit-4.0
3. Intel Math Kernel Library (MKL). Reference Manual/ https://software.intel.com/en-us/articles/mkl-reference-manual
4. MPI. http://www.mpi.org
5. OpenMP. V. 4.0. http://www.openmp.org/mp-documents/OpenMP4.0.pdf/
6. Химич А.Н., Молчанов И.Н., Попов А.В., Чистякова Т.В., Яковлев М.Ф. Параллельные алгоритмы решения задач вычислительной математики. Київ: Наук. думка, 2008. 248 с.
7. Хіміч О.М., Баранов А.Ю. Гібридний алгоритм розв'язування лінійних систем зі стрічковими матрицями прямими методами. Комп'ютерна математика. Зб. наук. праць. 2013. Вип. 2. С. 80–87.

8. Попов О.В., Рудич О.В. До розв'язування систем лінійних рівнянь на комп'ютерах гібридної архітектури. *Математичне та комп'ютерне моделювання.* Серія: Фізико-математичні науки: зб. наук. праць. 2017. Вип. 15. С. 158–164.
9. И.Н. Молчанов, А.В. Попов, А.Н. Химич Алгоритм решения частичной проблемы собственных значений для больших профильных матриц. *Кибернетика и системный анализ*. 1992. № 2. С. 141–147.
10. Химич А.Н., Попов А.В., Чистяков А.В. Гибридные алгоритмы решения алгебраической проблемы собственных значений с разреженными матрицами. *Кибернетика и системный анализ.* 2017. № 6. С. 132–146.

References

1. TOP 500 – 2017(11). http://www.TOP500.org/
2. CUDA TOOLKIT 4.0. http://developer.nvidia.com/cuda-toolkit-4.0
3. Intel Math Kernel Library (MKL). Reference Manual/ https://software.intel.com/en-us/articles/mkl-reference-manual
4. MPI. http://www.mpi.org
5. OpenMP. V. 4.0. http://www.openmp.org/mp-documents/OpenMP4.0.pdf/
6. A.N. Khimich, I.N. Molchanov, A V. Popov, T.V. Chistyakova, and M.F. Yakovlev, Parallel Algorithms for the Solving of Computational Mathematics Problems. [in Russian], Naukova Dumka, Kyiv (2008).
7. O.M. Khimich, A.Y. Baranov Hibrid Algorithm for Solving Linear Systems with Band Matrices by Direct Method. Computer Mathematics. Sb. sciences works. 2013. N 2. P. 80–87.
8. Popov O.V., Rudich O.V. On the Solving of Linear Systems on Hybrid-Architecture Computers. Mathematical and computer modeling. Series: Physics and Mathematics: Sb. sciences works. 2017. N 15. P. 158–164.
9. I.N. Molchanov, A.V. Popov, A.N. Khimich Algorithm to Solve the Partial Eigenvalue Problem for Large Profile Matrices. Cybernetics and Systems Analysis. 1992. N 2. P. 141–147.
10. Khimich A.N., Popov A.V., Chistyakov A.V. Hybrid Algorithms for Solving the Algebraic Eigenvalue Problem with Sparse Matrices. Cybernetics and Systems Analysis. 2017. N 6. P. 132–146.

Про авторів:

Попов Олександр Володимирович
кандидат фізико-математичних наук,
старший науковий співробітник,
старший науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України.
Кількість наукових публікацій в українських виданнях – близько 80.
h-індекс – 1.
https://orcid.org/0000-0002-1217-2534.

Рудич Ольга Василівна
науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України.
Кількість наукових публікацій в українських виданнях – 31,
https://orcid.org/0000-0003-3525-3174.

Чистяков Олексій Валерійович,
молодший науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України.
Кількість наукових публікацій в українських виданнях – 20.
h-індекс – 1.
https://orcid.org/0000-0001-6456-2094.

***Місце роботи авторів*:**

Інститут кібернетики імені В.М. Глушкова НАН України.
03187, Київ-187, проспект Академіка Глушкова, 40.
Тел.:(044) 526 1196.

E-mail: alex50popov@gmail.com,
olgarudich2016@gmail.com,
alexej.chystyakov@gmail.com