ПРОГРАМНО-ТЕХНІЧНІ КОМПЛЕКСИ

УДК 519.6

О.М. ХІМІЧ
Інститут кібернетики ім. В.М. Глушкова НАН України, Київ, Україна,
e-mail: *khimich505@gmail.com*.

О.В. ПОПОВ
Інститут кібернетики ім. В.М. Глушкова НАН України, Київ, Україна,
e-mail: *alex50popov@gmail.com*.

О.В. ЧИСТЯКОВ
Інститут кібернетики ім. В.М. Глушкова НАН України, Київ, Україна,
e-mail: *alexej.chystyakov@gmail.com*.

В.О. КОХАНОВСЬКИЙ
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна,
e-mail: *kokhanovstyy@gmail.com*.

РОЗВ'ЯЗУВАННЯ ЗАДАЧ НА ВЛАСНІ ЗНАЧЕННЯ У ЗМІННОМУ КОМП'ЮТЕРНОМУ СЕРЕДОВИЩІ СУПЕРКОМП'ЮТЕРІВ

Анотація. Запропоновано математичне забезпечення для дослідження та розв'язування на комп'ютері MIMD-архітектури з графічними процесорами алгебраїчної проблеми власних значень. До складу математичного забезпечення входять паралельні алгоритми та програми з функціями автоматичного адаптивного налаштування змінного комп'ютерного середовища (багаторівневий паралелізм, змінна топологія міжпроцесорних зв'язків, змішана розрядність, кешизація тощо) на виявлені в комп'ютері математичні властивості задачі та його архітектурні особливості для забезпечення достовірності результатів розв'язування і ефективного використання обчислювальних ресурсів.

Ключові слова: алгебраїчна проблема власних значень (АПВЗ), змінне комп'ютерне середовище, адаптивні алгоритми, метод спряжених градієнтів, метод ітерацій на підпросторі, багатоядерний комп'ютер MIMD-архітектури з графічними процесорами.

ВСТУП

Сьогодні в багатьох предметних галузях (ядерна енергетика, механіка, машинобудування, будівництво, електрозварювання, хімія, молекулярна біологія тощо) постійно виникають нові більш складні фізико-технічні об'єкти, конструкції та явища.

Поява нових надвеликих задач обчислювальної математики змушує постійно нарощувати потужності суперкомп'ютерів за рахунок ускладнення архітектури багатопроцесорних обчислювальних систем з розподіленою та ієрархічною спільною пам'яттю зі складною організацією багатопотокових і векторизованих операцій, із застосуванням десятків тисяч процесорів, різноманітних прискорювачів, видів кеш-пам'яті, систем розпаралелювання тощо.

З появою та постійним удосконаленням суперкомп'ютерів різної архітектури стають актуальними проблеми їхнього ефективного використання, забезпечення достовірності комп'ютерних результатів та підвищення рівня інтелектуальної інформаційної підтримки користувачів — фахівців у різних предметних галузях [1–4].

У цій статті розглядається розв'язання таких задач, а саме створення алгоритмічно-програмного забезпечення для дослідження та розв'язання алгебраїчної проблеми власних значень (АПВЗ) з функціями автоматичного адаптивного налаштування методу, комп'ютерного алгоритму та змінного комп'ютерного середовища (змінна топологія процесорів та міжпроцесорних зв'язків, багаторівневий паралелізм, багаторозрядна арифметика, кешизація обчислень тощо) на виявлені в комп'ютері математичні властивості задачі та його архітектурні особливості для забезпечення достовірності результатів розв'язування під час ефективного використання обчислювальних ресурсів [5–7].

Багато практичних задач, наприклад задача розрахунку стійкості конструкцій, що включає умови неперервності векторів напруги та зміщень на межах області [8], можуть бути представлені у вигляді задач на власні значення у нескінченновимірному просторі. Математичні моделі таких задач описуються системами диференціальних рівнянь або різницевими рівняннями, розв'язання яких полягає в обчисленні власних значень і власних векторів матриць, що зазвичай мають розріджену структуру (кількість ненульових елементів матриць становить kn, де $k << n$, а n — порядок матриці) та надвеликі порядки (до десятків мільйонів).

На основі досліджень задач, що зводяться до розв'язання АПВЗ, створено такі ітераційні адаптивні алгоритми в змінному комп'ютерному середовищі:

- алгоритми поперемінно-трикутного методу для дослідження та розв'язування часткової узагальненої АПВЗ (мінімальне власне значення та відповідний власний вектор) симетричних додатно визначених стрічкових та розріджених матриць довільної структури [7];
- алгоритми методу спряжених градієнтів для дослідження та розв'язування часткової узагальненої АПВЗ (мінімальне власне значення та відповідний власний вектор) симетричних додатно визначених стрічкових та розріджених матриць довільної структури [9, 10];
- алгоритми методу ітерацій на підпросторі для дослідження та розв'язування часткової узагальненої АПВЗ (декілька мінімальних власних значень та відповідних власних векторів) симетричних додатно визначених стрічкових та розріджених матриць довільної структури [9–12].

ЗМІННЕ КОМП'ЮТЕРНЕ СЕРЕДОВИЩЕ

Сучасні паралельні суперкомп'ютери можуть бути різної архітектури. Упродовж десятків років широко використовують комп'ютери MIMD-архітектури, в якій з'єднані загальною шиною обчислювальні вузли складаються з певної кількості процесорів з розподіленою пам'яттю та (або) ядер із спільною пам'яттю. Останнім часом активно розробляють комп'ютери гібридної архітектури — паралельні комп'ютери з різними співпроцесорами. Серед них найбільш ефективними є комп'ютери MIMD-архітектури (CPU) з графічними прискорювачами (GPU).

Створюючи алгоритми для розв'язання задач на сучасних суперкомп'ютерах, потрібно враховувати багаторівневу модель розпаралелювання, наявність комп'ютерної пам'яті різних видів та обсягів, кількість і типи процесорних елементів та особливості зв'язків між ними тощо [5–7].

Багаторівнева модель паралельних обчислень. На сучасних суперкомп'ютерах передбачено два основні рівні паралелізму: верхній, на якому паралельно виконуються макрооперації (підзадачі — логічно незалежні частини алгоритмів), та нижній, де розпаралелюється виконання кожної макрооперації.

Перший (верхній) рівень моделі паралельних обчислень (MIMD-модель) — паралелізм процесів (process level parallelism, PLP), тобто процеси паралельно виконують підзадачі, використовуючи як розподілену, так і спільну пам'ять CPU, синхронізуючи обчислення та обміни даними. Для розпаралелювання процесів на розподіленій пам'яті найбільш ефективною є інтерфейс MPI [13], а на спільній пам'яті — OpenMP [14].

Другий (нижній) рівень (SIMD-модель) — паралелізм потоків (thread level parallelism, TLP) — розпаралелювання макрооперацій з використанням декількох потоків і спільної пам'яті. У цьому випадку кожна макрооперація верхнього рівня розпаралелюється між деякою кількістю потоків (ниток) на ядрах CPU із спільною пам'яттю за допомогою засобів OpenMP, а також на співпроцесорах, наприклад GPU, під час використання технології NVIDIA CUDA [15].

У разі використання розподіленої пам'яті виникають певні проблеми з обмінами даних між процесами — операціями, які залежать від реалізації (встановленої на комп'ютері) стандарту MPI та тривалість яких може значно перевищувати тривалість арифметичних операцій, а також операцій звернення до пам'яті. Отже, під час створення алгоритмів для розв'язання задач потрібно передбачати таке розміщення даних у пам'яті процесорів, за якого співвідношення пересилання даних та арифметичних операцій, що одночасно виконуються, буде збалансованим і мінімізуватиме загальний час виконання програми. З іншого боку, середовище MPI створює достатньо сприятливі умови для асинхронного виконання обмінів даними (під час виконання арифметичних операцій).

За використання спільної пам'яті та середовища OpenMP не виникають проблеми з обмінами даних між потоками, але створюються проблеми з коректним використанням даних, для уникнення яких потрібні додаткові синхронізації та обміни 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 асинхронного виконання різних макрооперацій створює значні труднощі. У цьому середовищі для ефективної реалізації паралельних обчислень також потрібно передбачати рівномірне (збалансоване) завантаження використовуваних потоків. Отже, якщо кількість інформаційних зв'язків (комунікаційних взаємодій) між підзадачами є достатньо великою (порівняно з виконанням арифметичних операцій) або паралельно мають виконуватися однакові підзадачі з різними даними, то доцільно реалізовувати відповідні алгоритми у середовищі OpenMP. Якщо в алгоритмі кількість інформаційних зв'язків є мінімальною, а виконуються переважно арифметичні операції, то використання середовища MPI може бути ефективним.

Основними проблемами паралельних обчислень для комп'ютерів гібридної архітектури є узгодження розподілу обчислювальних ресурсів між CPU (ядрами процесорів) та GPU (графічними прискорювачами), а також оптимізація комунікаційних витрат між CPU та GPU. Для підвищення ефективності адаптивних алгоритмів потрібно використовувати асинхронне виконання операцій (багатопотокове розпаралелювання на GPU та копіювання даних між CPU і GPU), а також взаємодію між декількома GPU з одним CPU (застосовуючи, наприклад, механізм потоків cudaStream) [15].

Змінна топологія міжпроцесорних зв'язків. Ефективність розв'язування обчислювальних задач на паралельних комп'ютерах значною мірою залежить від особливостей зв'язків між процесорами та між ядрами всередині процесорів. Існує декілька рівнів таких зв'язків: між обчислювальними вузлами, між процесорами одного вузла та між ядрами одного процесора.

Під час використання розподіленої пам'яті виникають певні проблеми з обмінами даних між процесами, які за тривалістю можуть значно перевищувати арифметичні операції та операції звернення до пам'яті. Отже, потрібно передбачати такі комунікаційні зв'язки між обчислювальними елементами CPU, за яких співвідношення їхніх пересилань та арифметичних операцій, що одночасно виконуються, буде збалансованим і мінімізуватиме загальний час виконання програми. Функції середовища MPI надають можливість програмно створювати ефективні віртуальні топології з визначеної кількості процесів на час виконання задачі [13].

Розрізняють такі види топологій: одновимірна (лінійний масив), двовимірна (кільце, зірка, дерево, ґрати), тривимірна (повнозв'язна топологія, хордальне кільце) та гіперкубічна [4].

Комп'ютерне кешування. Процесори сучасних суперкомп'ютерів з багаторівневою комп'ютерною моделлю мають кеш-пам'ять — надшвидку статичну пам'ять. У ній зберігається деяка кількість останніх використаних інструкцій та даних так, що цикли і операції з масивами будуть виконуватися значно швидше. Тобто кеш є буфером, в який завантажуються дані, і, незважаючи на невеликий обсяг (близько 4–16 Мбайт), він забезпечує значне зростання продуктивності обчислень.

Найчастіше сучасні комп'ютери мають два або три рівні кеш-пам'яті. Перший рівень (L1) — найбільш швидкий, який працює безпосередньо з ядром процесора і є буфером між процесором та кеш-пам'яттю другого рівня. Другий рівень (L2) масштабніший за перший, але має менші «швидкісні характеристики». Відповідно він є буфером між рівнями L1 та L3. Третій рівень (L3) повільніший за попередні, проте математичні операції виконуються набагато швидше, ніж на оперативній пам'яті. На відміну від L1 та L2, які розподіляються між ядрами, цей рівень є загальним для всього процесора.

Таким чином, швидкодію обчислювальних алгоритмів та програм можна значно підвищити, якщо розробляти їх з урахуванням наявної кеш-пам'яті процесорів. Дослідження показали, що для розв'язання задач лінійної алгебри на паралельних комп'ютерах, зокрема і гібридної архітектури, для ефективного використання обчислювальних ресурсів (кеш-пам'яті на CPU і швидкодії багатопоточного виконання однотипних арифметичних операцій на GPU) ефективнішими є блочні алгоритми [4].

У процесах, що виконуються на CPU, розмір блока матриці можна узгодити з розміром кеш-пам'яті, а на процесах GPU вихідний масив матриці розбити на блоки так, щоб їхня довжина відповідала розміру блока shared-пам'яті, на якій виконуються обчислення. Зазвичай розмір блока матриці на GPU є кратним 16 [6, 7].

АВТОМАТИЧНЕ НАЛАШТУВАННЯ ЕФЕКТИВНОГО КОМП'ЮТЕРНОГО СЕРЕДОВИЩА В АДАПТИВНИХ АЛГОРИТМАХ

У результаті проведеного аналізу архітектурних та технологічних особливостей багатоядерних комп'ютерів з графічними прискорювачами, багаторівневої моделі паралельних обчислень, функціонального призначення розглянутих адаптивних ітераційних алгоритмів було сформульовано такі вимоги до реалізації автоматичного налаштування ефективного змінного комп'ютерного середовища:

- декомпозиція вихідної задачі на окремі підзадачі, які можуть бути реалізовані значною мірою незалежно одна від одної;
- визначення, які підзадачі доцільніше виконувати на CPU, а які — на GPU, та встановлення інформаційних залежностей між ними;
- створення ефективної логічної топології MIMD-комп'ютера з потрібної кількості процесорних ядер CPU, тобто з виконуваних на них обчислювальних процесів;

• рівномірне завантаження процесів, що використовуються, та синхронізація обмінів між ними;

• виконання на GPU математичних операцій, які потребують найбільших затрат комп'ютерного часу;

• мінімізація обмінів між процесами CPU та між CPU і GPU;

• масштабованість алгоритму — забезпечення можливості ефективно розв'язувати задачі з використанням різної кількості процесів;

• ефективні способи оброблення розріджених матриць.

На основі сформульованих вимог, а також теоретичного та експериментального дослідження ефективності адаптивних алгоритмів для розріджених матриць різного вигляду реалізовано технологію автоматичного визначення змінного комп'ютерного середовища та структури даних для розв'язування обчислювальних задач на гібридних комп'ютерах [6–12].

Як відомо, для оцінки якості паралельного алгоритму використовують коефіцієнти прискорення S_p та ефективності E_p [4]:

$$S_p = T_1 / T_p,\ E_p = S_p / p,$$

де p — кількість використаних для обчислень процесів, T_1 — час розв'язання задачі одним процесом, T_p — час розв'язання тієї ж задачі p-паралельними процесами.

Для гібридних комп'ютерів, якщо позначити T_1 час розв'язування задачі на архітектурі з використанням одного CPU та одного GPU, T_p — час розв'язування цієї ж задачі у разі використання pCPU та pGPU, то час розв'язання задачі можна обчислити за формулами [6–12]:

$$T_1 = Ot_g,\ T_p = Ot_g + M_1 t_{opg} + M_2 t_{opp} + Q_1 t_{cpg} + Q_2 t_{cpp}.$$

Тут O — кількість виконаних алгоритмом операцій; t_g — середній час виконання однієї арифметичної операції на GPU; t_{opg} — час обміну інформацією між CPU та GPU; t_{opp} — час обміну одним машинним словом між двома процесами CPU; t_{cpg} — час для встановлення зв'язку між CPU і GPU; t_{cpp} — час для встановлення зв'язку між двома процесами на CPU; M_i та Q_i — відповідно кількість обмінів та синхронізацій між CPU та GPU на i-й ітерації ($i = 1, 2, \ldots$).

З використанням таких формул для кожного розробленого адаптивного алгоритму отримано апріорні оцінки коефіцієнтів прискорення та ефективності для різних паралельних моделей обчислень. Крім того, проведено експериментальне дослідження ефективності створених алгоритмів на паралельних комп'ютерах різної архітектури для розв'язування низки тестових задач з розрідженими матрицями різних порядків та різної структури [6–12]. Отримані в експериментах показники ефективності добре узгоджуються з теоретичними оцінками.

Комп'ютерне дослідження структури розрідженої матриці та ідентифікація її математичних властивостей. Проведення математичного моделювання процесів, які виникають в аналізі складних багатокомпонентних середовищ, показало, що вони часто зводяться до розв'язання АПВЗ розріджених матриць надвеликих порядків та невизначеної структури. У наведених адаптивних алгоритмах у цьому випадку передбачається використання відповідних алгоритмічно-програмних засобів для автоматичного розпізнавання апріорі невідомої структури розрідженої матриці методами машинного навчання і штучних нейронних мереж, за допомогою яких здійснюється візуалізація матриці на основі класифікації об'єктів та дослідження вигляду («портрету») матриці [16].

Також у разі потреби може бути проведена автоматична ідентифікація математичних властивостей матриць на основі багаторозрядної арифметики.

Вихідні дані математичних моделей більшості прикладних задач, які доводиться розв'язувати на комп'ютерах, задаються наближено. Крім того, під час введення даних в комп'ютер здійснюється їхнє округлювання до обчислювальної розрядності. Одним із способів запобігання похибок під час округлювання даних в алгоритмах, які призначені для розв'язання задач лінійної алгебри, є проведення комп'ютерного дослідження обумовленості введених матриць. Проте, в комп'ютерних обчисленнях критерій хорошої або поганої обумовленості матриці є величиною відносною. Наприклад, одна й та сама матриця може класифікуватися для однієї довжини мантиси машинного слова як добре обумовлена, а для іншої — як погано обумовлена [4].

Як приклад можна навести задачу чисельного моделювання напружено-деформованого стану нескінченного напівпростору з двошаровим включенням [17].

У багатьох галузях сучасної техніки широко застосовують композиційні матеріали, в яких армованими елементами є сферичні частинки різних діаметрів. Потреба у визначенні характеристик таких матеріалів зумовлює створення математичних моделей, що описують процеси взаємодії сферичного наповнювача з середовищем (матрицею), в яких першорядне значення має обчислення напружено-деформованого стану в системі «сферичний наповнювач–полімерна матриця». Розв'язок задачі шукають у зміщеннях з використанням розкладання Папковича–Нейбера [18].

Особливістю цієї задачі та методу її розв'язання є те, що в залежності від геометричних розмірів включень та їхнього розташування відносно краю, змінюється значення числа обумовленості матриці комп'ютерної моделі задачі від помірного до критичного, коли матриця стає машинно виродженою. Отримати машинний розв'язок СЛАР, що зберігає фізичний сенс, неможливо без дослідження узгодженості математичних властивостей комп'ютерної моделі задачі та довжини машинного слова. Для подальшого дослідження комп'ютерної моделі задачі доцільно збільшувати комп'ютерну розрядність [19, 20].

Адаптивні алгоритми дослідження невідомих математичних властивостей машинної моделі матриці задачі можна проводити, використовуючи багаторозрядну арифметику, за допомогою спеціальних алгоритмічно-програмних засобів [19, 20].

Зазначимо, що під час створення адаптивних алгоритмів для розв'язання часткової АПВЗ змішану розрядність було також використано для зменшення обчислювальних ресурсів на комп'ютерах з невеликою оперативною пам'яттю. Як відомо, під час математичного моделювання практичних задач переважна більшість обчислень проводиться з подвійною розрядністю. Водночас використання даних задачі у форматі з одинарною точністю дає змогу зберігати їх вдвічі більше на кожному рівні ієрархії пам'яті включно з кеш-пам'яттю та регістровою пам'яттю. Як показали дослідження, застосування змішаної розрядності дає можливість скорочувати час математичного моделювання та комп'ютерні ресурси, не втрачаючи точності отримуваних результатів розв'язування [21]. В адаптивному алгоритмі методу ітерацій на підпросторі для розв'язання АПВЗ блочно-діагональних матриць з обрамленням однією з підзадач є розв'язання СЛАР, що потребує найбільшого часу та комп'ютерних ресурсів, виконується дрібноплитковим адаптивним алгоритмом LL^{T}-розкладання над невеликими блоками матриць з використанням змішаної розрядності (одинарної та подвійної) [11].

Структурна регуляризація та декомпозиція розріджених матриць. Для ефективного використання комп'ютерних ресурсів та часу розв'язування в процесі створення алгоритмів для розріджених матриць важливо передбачати їхнє зведення до регулярного вигляду та зберігання ненульових елементів, застосовуючи стислі формати даних. Існує низка методів, які дають змогу регулювати заповнення матриці під час реалізації розв'язування задачі на комп'ютері. Розроблено як методи загального призначення [22], так і орієнтовані на матриці конкретного вигляду [23–25].

Розглянемо застосування деяких способів оброблення розріджених матриць на прикладі двох адаптивних алгоритмів для розв'язування на гібридних комп'ютерах часткової АПВЗ методом спряжених градієнтів для блочно-діагональних матриць з обрамленням та методом ітерацій на підпросторі для стрічкових матриць.

Зазначимо, що обчислювальні схеми поперемінно-трикутного методу та методу спряжених градієнтів збігаються в реалізації основних підзадач [7], тому тут розглядається адаптивний алгоритм методу спряжених градієнтів.

Адаптивний алгоритм методу спряжених градієнтів. Цей алгоритм призначений для знаходження мінімального власного значення АПВЗ вигляду

$$Ax = \lambda x \tag{1}$$

неявним методом спряжених градієнтів, реалізується за формулами [4, 9, 10]:

$$x^{k+1} = \begin{cases} x^k + a_k p^k, & k \neq N-1, 2N-1, \dots, \\ \dfrac{x^k - a_k p^k}{||\, x^k - a_k p^k ||_2}, & k = N-1, 2N-1, \dots \end{cases} \tag{2}$$

Тут маємо

$$p^k = Z(x^k) f(x^k) - \beta_k p^{k-1}, \; f(x) = [A - \mu(x) I] x / ||\, x ||_2, \tag{3}$$

$$\beta_k = \begin{cases} 0, & k = 0, 2N, \dots, \\ \dfrac{((Z(x^k) f(x^k)), ([A - \mu(x^k) I]) p^{k-1}))}{(p^{k-1}, [A - \mu(x^k) I]) p^{k-1})}, & k = 1, 2, 3, \dots, \end{cases}$$

$$\mu(x) = \frac{(Ax, x)}{(x, x)}.$$

Параметр a_k може бути вибраний як точка локального мінімуму функціонала $\mu(x^k - ap^k)$, N — момент відновлення, $Z(x^k)$ — деяка матриця, яку вибирають з умов прискорення збіжності ітераційного процесу. Наприклад, Z можна вибрати на основі методу верхньої симетричної релаксації. У цьому випадку, якщо вихідну матрицю представити у вигляді $A = I - L - L^{\mathrm{T}}$ через одиничну, нижню та верхню трикутні матриці, то матриця Z буде мати вигляд

$$Z(x) = \omega(2 - \omega)(I - \omega \hat{L}^{\mathrm{T}}(x))^{-1} (I - \omega \hat{L}(x))^{-1}, \tag{4}$$

де $\hat{L}(x) = \dfrac{1}{1 - \mu(x)} L$, ω — параметр релаксації, $\omega \in (1, 2)$.

Ітераційний процес закінчується за критерієм $\dfrac{|\, \mu^{(k+1)} - \mu^{(k)} |}{\mu^{(k)}} \leq eps$.

Згідно з теоремою [9], якщо $m ||\, y||_2^2 \leq (B(x) y, y) \leq M ||\, y||_2^2, x, \; y \in R_n, 0 < m \leq$ $\leq M < +\infty$ та $N < +\infty$, то ітераційний процес збігається для будь-якого початкового наближення $x^{(0)}$, причому $\lim \mu(x^k) = \lambda_i$, $\lim x^k = v_i$, де λ_i, v_i — одне із власних значень та відповідний йому власний вектор.

Для ефективного використання розріджених матриць довільного вигляду на гібридному комп'ютері матриця A задачі (1) зводиться методом паралельних перерізів до блочно-діагонального вигляду з обрамленням [26]:

$$\tilde{A} = P^{\mathrm{T}} A P = \begin{pmatrix} D_1 & 0 & 0 & \dots & 0 & C_1 \\ 0 & D_2 & 0 & \dots & 0 & C_2 \\ 0 & 0 & D_3 & & 0 & C_3 \\ \vdots & \vdots & & \ddots & & \vdots \\ 0 & 0 & 0 & & D_{p-1} & C_{p-1} \\ C_1 & C_2 & C_3 & \dots & C_{p-1} & D_p \end{pmatrix}, \tag{5}$$

де P — матриця перестановок, а блоки D_i і C_i зберігають розрідженість. Природним для методу паралельних перерізів є те, що порядок останнього діагонального блока матриці набагато менший, ніж порядки інших діагональних блоків.

У результаті застосування методу паралельних перерізів вихідна задача (1) зводиться до еквівалентної задачі

$$\widetilde{A}y = \lambda y,$$

де $y = P^{\mathrm{T}}x$, $\widetilde{A}$ — блочно-діагональна матриця з обрамленням. Оскільки тут використано перетворення подібності, власні значення отриманої матриці $\widetilde{A}$ і вихідної A збігаються.

З урахуванням структури отриманої матриці $\widetilde{A}$ (5) було використано розподіл блоків між процесами на CPU, причому кількість блоків визначалась відповідно до кількості використаних процесів, а саме процеси з номерами $1 \le i < p$ зберігають блоки D_i та C_i, а процес з номером p зберігає блок D_p.

Отже, будемо розв'язувати задачу (1) з отриманою блочно-діагональною матрицею з обрамленням. Реалізація алгоритму методу спряжених градієнтів на основі методу симетричної верхньої релаксації для розрідженої блочно-діагональної матриці з обрамленням визначається згідно з (2)–(4) в основному розв'язуванням двох систем: $(I - \omega\widetilde{\widetilde{L}}(x))y = z(x)$ з нижньою трикутною матрицею та $(I - \omega\widetilde{\widetilde{L}}^{\mathrm{T}}(x))w = y(x)$ з верхньою трикутною матрицею, де $z(x) = \omega(2-\omega)f(x)$, $w^k = p^k - \beta_k p^{k-1}$, k — номер ітерації.

Розріджена блочно-трикутна матриця $\widetilde{\widetilde{L}}$ з обрамленням має вигляд:

$$\widetilde{\widetilde{L}} = \begin{pmatrix} \widetilde{L}_1 \\ \widetilde{L}_2 \\ \widetilde{L}_3 \\ \vdots \\ \widetilde{L}_{p-1} \\ \widetilde{L}_p \end{pmatrix} = \begin{pmatrix} \widetilde{D}_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \widetilde{D}_2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \widetilde{D}_3 & \dots & 0 & 0 \\ \vdots & \vdots & & \ddots & & \vdots \\ 0 & 0 & 0 & \dots & \widetilde{D}_{p-1} & 0 \\ C_1 & C_2 & C_3 & \dots & C_{p-1} & \widetilde{D}_p \end{pmatrix}.$$

Тут розглядається багатоядерний MIMD-комп'ютер з логічною топологією між CPU-процесами «гіперкуб» та відповідними GPU-процесорами. Процес CPU(q) виконується на ядрі CPU з логічним номером q ($q = 1, 2, \dots, p$) з використанням відповідного GPU(q), де p — кількість задіяних процесів CPU та процесорів GPU.

Адаптивний алгоритм методу ітерацій на підпросторі. Теоретичні аспекти цього алгоритму викладено в [25]. Він призначений для обчислення r мінімальних власних значень і відповідних до них власних векторів задачі

$$Ax = \lambda Bx, \tag{6}$$

де A, B — симетричні стрічкові матриці порядку n, матриця A — додатно визначена.

Цей метод є узагальненням методу обернених ітерацій і полягає у побудові для задачі (6) послідовності підпросторів E_t ($t = 1, 2, \dots$), яка збігається з підпростором E_∞, що містить шукані власні вектори [4, 25]. На t-й ітерації обчислюється ортогональний базис підпростору E_t і, якщо досягнуто потрібну точність наближеного розв'язку, визначаються шукані власні пари.

Таким чином, реалізація методу ітерацій на підпросторі задачі (6) зводиться до виконання для $t = 1, 2, \dots$ таких кроків:

- знаходження розв'язку СЛАР

$$AX_t = Y_{t-1}; \tag{7}$$

- обчислення проєкції матриці A на підпростір E_t

$$A_t = X_t^{\mathrm{T}} Y_{t-1} \equiv X_t^{\mathrm{T}} AX_t; \tag{8}$$

- обчислення прямокутної матриці

$$W_t = BX_t; \tag{9}$$

- обчислення проєкції матриці B на підпростір E_t

$$B_t = X_t^{\mathrm{T}} W_t \equiv X_t^{\mathrm{T}} BX_t; \tag{10}$$

- розв'язування повної проблеми власних значень для проєкцій

$$A_t Z_t = B_t Z_t \Lambda_t, \text{ де } \Lambda_t = \mathrm{diag}\,(\lambda_i); \tag{11}$$

- обчислення наближення

$$Y_t = W_t Z_t. \tag{12}$$

Якщо після t ітерацій виконуються умови закінчення ітераційного процесу, наприклад $\left| \dfrac{\lambda_i^{(t)} - \lambda_i^{(t-1)}}{\lambda_i^{(t)}} \right| \le \varepsilon$, то проводиться додаткова ітерація і наближені розв'язки задачі (6) набувають власних значень $\lambda_i^* = \lambda_i^{(c+1)}$, де $i = 1, 2, \ldots, r$, та мають власні вектори — перші r стовпчиків матриці $X^* = X_{c+1} Z_{c+1}$. Мова йде про те, що власні значення упорядковані за зростанням, тобто $0 < \lambda_1 \le \lambda_2 \le \ldots \le \lambda_r \le \ldots$

Ітераційний процес збігається лінійно, причому швидкість збіжності λ_i визначається відношенням λ_q / λ_1, де q — розмір підпростору E_q, що ітерується. У послідовних реалізаціях алгоритму рекомендується вибирати $q = \min(2r, r+8)$ [26].

Аналізуючи схему реалізації методу ітерацій на підпросторі, можна зазначити, що найбільш витратною операцією щодо використання обчислювальних ресурсів та комп'ютерного часу є розв'язування СЛАР алгоритмом LL^{T}-розкладання стрічкової матриці. Оскільки в обчислювальній схемі (7)–(12) на кожній ітерації виконується розв'язування СЛАР (7) з однією і тією ж матрицею A, то LL^{T}-розкладання доцільно виконати до початку ітераційного процесу. Тоді на кожній ітерації розв'язується задача (7) із факторизованою матрицею, що значно зменшує час виконання обчислень.

Отримавши в комп'ютері розв'язок задачі, необхідно дослідити, що знайдено всі потрібні власні значення та відповідні власні вектори. Адже під час обчислення k мінімальних власних значень може виникнути ситуація, коли замість одного з них обчислено $(k+1)$-е власне значення. Зокрема, подібні ситуації можуть виникати у випадку, коли початковий підпростір E_0 виявиться B-ортогональним до принаймні одного з шуканих власних векторів. До того ж, більш складно забезпечити апріорну B-ортогональність початкового підпростору до всіх шуканих векторів, розв'язуючи задачу на MIMD-комп'ютері або на гібридному комп'ютері. Адже в процесі реалізації паралельного алгоритму методу ітерацій на підпросторі матриця векторів, на які натягнуто початковий підпростір, формується з розподілених її частин між паралельними обчислювальними пристроями. Виявити ситуацію, коли через ортогональність до початкового підпростору не знайдено одну (або більше) з шуканих власних пар, можна за властивістю послідовності Штурма [4] для задачі (7).

В адаптивному алгоритмі методу ітерацій на підпросторі, як і в адаптивному алгоритмі методу спряжених градієнтів, розглядаємо гібридний комп'ютер, архітектура якого складається з багатоядерного комп'ютера MIMD-архітектури з топологією комунікаційних зв'язків між процесами — «гіперкуб» та декількох графічних процесорів. Тобто маємо гібридний комп'ютер з pCPU та pGPU, p — загальна кількість процесів, що виконуються на центральному процесорі (CPU).

Для розв'язання задачі (7) на гібридному комп'ютері матриця A розбивається на квадратні блоки $A_{I,J}$ порядку s. Елементи головної діагоналі та нижнього або верхнього трикутника (в залежності від використаних алгоритмів) ненульових блоків цієї стрічкової симетричної матриці розподіляються між процесами CPU відповідно до одномірної блочно-циклічної схеми [4, 9]. За цією схемою блок $A_{I,J}$ зберігається в процесі на CPU з логічним номером $(I + l) \bmod p$ (результат операції $k \bmod j$ — залишок від ділення k на j, $-1 \le l \le p-2$ — зсув, зазвичай $l = -1$).

У результаті виконання LL^{T}-розкладання матриці A блоки нижньої трикутної матриці L або верхньої трикутної матриці L^{T} будуть розподілені блочно-циклічним способом [27]. Така ж схема декомпозиції між процесорами використовується для елементів матриці B та прямокутних матриць ітерованих векторів X_t, Y_t, W_t. При цьому достатньо розподіляти та зберігати лише ненульові елементи матриці B у такій послідовності: піддіагональні, діагональні та наддіагональні. Це значно спрощує алгоритм перемноження такої матриці на прямокутну матрицю, несуттєво збільшуючи загальний обсяг даних. У пам'яті процесорів GPU потрібно зберігати (відповідно до розподілу між процесами на CPU) копії тих блоків матриць A, B, X_t, Y_t, W_t, які використовуються для виконання відповідних математичних операцій.

З аналізу представленого адаптивного алгоритму бачимо, що він зводиться до виконання різних обчислень над розрідженими матрицями великих порядків. Ефективність обчислень на CPU можна забезпечити паралельним виконанням потоків інструкцій на ядрах та процесорах, а також використанням великої кеш-пам'яті, а на GPU — розпаралелелюючи на безліч потоків матрично-векторні обчислення.

Експериментальне дослідження адаптивних алгоритмів для розв'язування часткової АПВЗ. Дослідження цих алгоритмів для розв'язування часткової АПВЗ розріджених симетричних додатно визначених матриць різної структури методом спряжених градієнтів та методом ітерацій на підпросторі проводилось у змінному комп'ютерному середовищі гібридного комп'ютера СКІТ-4. Його технічні характеристики: CPU — Intel(R) Xeon(R) CPU E5-26700, тактова частота — 2.60 GHz, швидкість — 8 GT/s, кеш-пам'ять — 20 MB; у вузлі: 2 CPU по 8 ядер + Hyperthrefding = 32 ядра, Max Memory Size — 384 GB; GPU — Nvidia Tesla M2050, пам'ять — 3 GB, пікова продуктивність — 515 Gflops [28]. Використовувалися симетричні розріджені матриці різних порядків з Флоридської колекції [29] (табл. 1).

На рис. 1 показано прискорення алгоритму методу спряжених градієнтів під час знаходження найменшого власного значення розріджених матриць різного порядку.

Як бачимо з графіків (див. рис 1), отримане прискорення обчислень під час розв'язання задачі адаптивним алгоритмом на комп'ютерній архітектурі 1CPU + 1GPU порівняно з послідовною версією алгоритму (1CPU) буде від 6.5 до 7.5 раз більше, а у випадку використання архітектури 8CPU + 8GPU — до 45 раз.

На рис. 2 наведено результати дослідження залежності прискорення адаптивного алгоритму методу спряжених градієнтів від розміру блоків у приведенні вихідної матриці до вигляду блочно-діагональної матриці з обрамленням.

Таблиця 1. Тестові розріджені матриці з Флоридської колекції

Назва задачі	Проблемна галузь	Порядок матриці (напівширина стрічки)
G2_circuit	Structural problem	150102 (65956)
Bone010	Model reduction problem	986703 (26572)
Emila_923	Structural problem	923136 (34175)

Розв'язувалась часткова АПВЗ для матриць Bone010 порядку 986703 та з кількістю ненульових елементів 47851783.

З рис. 2 видно, що вибір розміру блока, узгодженого з кеш-пам'яттю центрального процесора та з кількістю використаних процесів, сприяє підвищенню прискорення розв'язання задачі. На одновузловій гібридній архітектурі було отримано найбільше прискорення за збільшення розмірів блока матриць до 512 і більше, а на багатовузловій гібридній архітектурі — за розміру блока 1024 і більше.

Прискорення адаптивного алгоритму методу ітерацій на підпросторі для різних матриць показано на графіках рис. 3. Тут розроблений адаптивний алгоритм для стрічкових матриць добре масштабований, прискорення за збільшення кількості процесів CPU та відповідних GPU значно зростає для всіх наведених

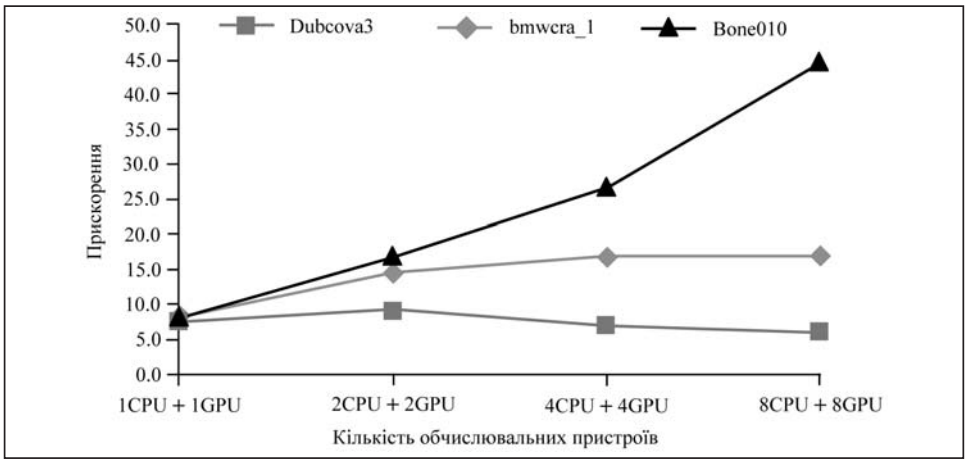

Рис. 1. Прискорення алгоритму методу спряжених градієнтів

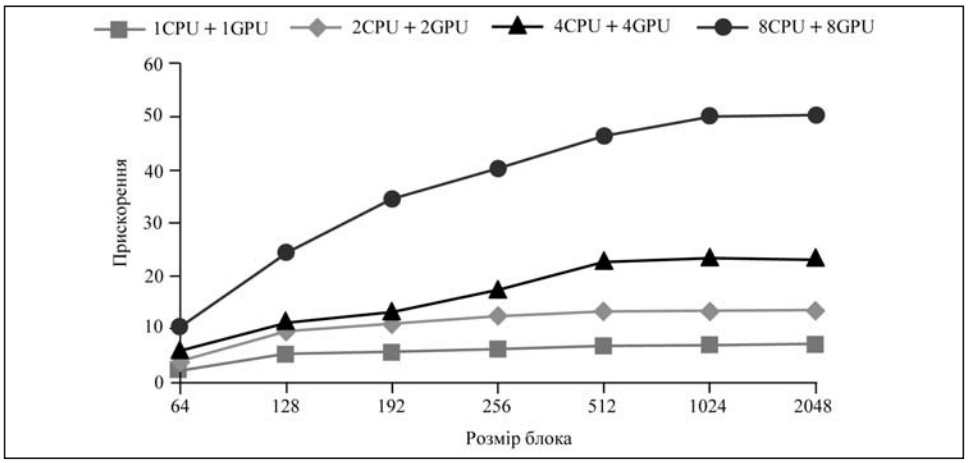

Рис. 2. Залежність прискорення алгоритму методу спряжених градієнтів від розміру блоків

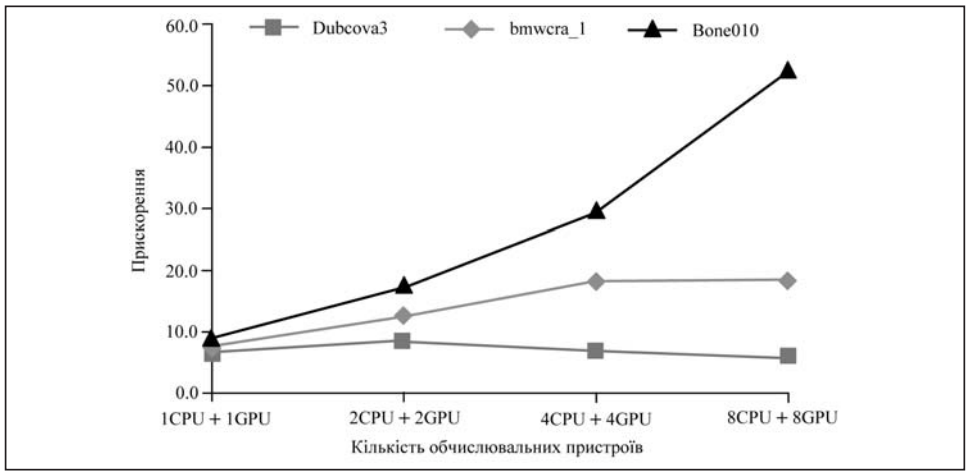

Рис. 3. Прискорення адаптивного алгоритму методу ітерацій на підпросторі

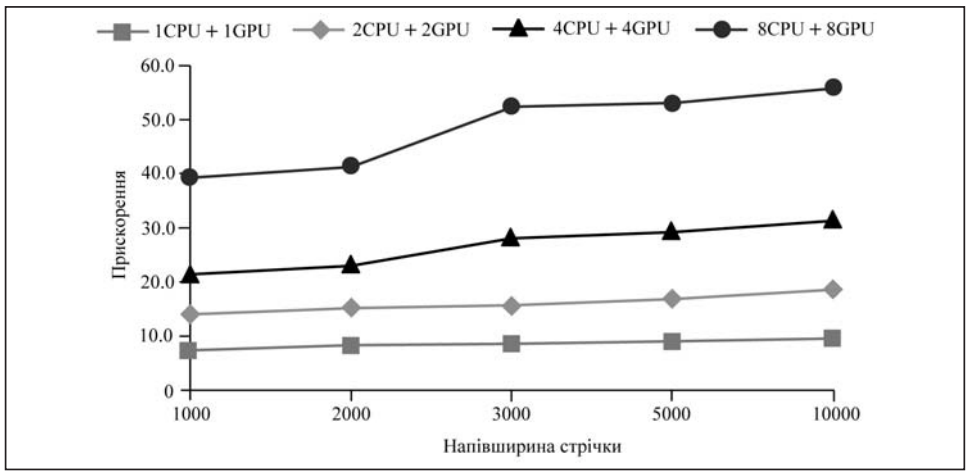

Рис. 4. Залежність прискорення алгоритму методу ітерацій на підпросторі від напівширини стрічки матриці

задач. Крім того, можна зазначити, що для матриць невеликого розміру (Dubcova3) у разі виходу за межі обчислювального вузла (коли кількість використаних GPU більша двох) наступає насичення процесу обчислювальними ресурсами. Водночас для матриць великого порядку bmwcra_1 та Bone010 зі збільшенням кількості CPU та GPU прискорення зростає.

На рис. 4 наведено залежність прискорення адаптивного алгоритму від напівширини стрічки матриці та кількості використаних GPU. Розв'язувалася часткова узагальнена АПВЗ стрічкових симетричних матриць порядку 250000 з різною напівшириною стрічки. Матриці отримано дискретизацією методом скінченних елементів змішаної крайової задачі для оператора Лапласа в прямокутному паралелепіпеді. З графіків (див. рис. 4) бачимо, що за збільшення напівширини стрічки матриці від 3000 і більше отримано найбільше прискорення на багатовузловому гібридному комп'ютері (кількість GPU більша двох).

У випадку використання матриці з меншою напівшириною стрічки отримане прискорення набагато менше. Це зумовлено недостатньою завантаженістю обчислювальних пристроїв та великою кількістю міжпроцесорних обмінів.

Використання змінної розрядності для отримання достовірних розв'язків АПВЗ. Як відомо (див., наприклад, [25]), у випадку кратних власних значень встановлено, що кожний наближений власний вектор (який відповідає кратним власним значенням) є наближенням до вектора, що лежить в інваріантному підпросторі, базисом якого є власні вектори, які відповідають кратним власним значенням вихідної задачі. Якщо власні значення не є кратними, а близькими, то кожний обчислений власний вектор (який відповідає одному з близьких власних значень) є наближенням до одного з власних векторів вихідної задачі.

Апарат багаторозрядної арифметики дає змогу ідентифікувати випадки кратних і «патологічно» близьких власних значень. Для прикладу розглянемо АПВЗ симетричної матриці 3-го порядку

$$A=\begin{pmatrix} \frac{3}{2}+\frac{2w_1+8w_2}{15} & \frac{3}{\sqrt{6}}-\frac{(3w_1+2w_2)}{5\sqrt{6}} & \frac{1}{\sqrt{2}}+\frac{(w_1-2w_2)}{3\sqrt{2}} \\ \frac{3}{\sqrt{6}}-\frac{(3w_1+2w_2)}{5\sqrt{6}} & 2+\frac{9w_1+w_2}{20} & \frac{\sqrt{3}}{2}-\frac{3w_1-w_2}{4\sqrt{3}} \\ \frac{1}{\sqrt{2}}+\frac{(w_1-2w_2)}{3\sqrt{2}} & \frac{\sqrt{3}}{2}-\frac{3w_1-w_2}{4\sqrt{3}} & 1+\frac{5(w_1+w_2)}{12} \end{pmatrix}.$$

Для $w_1=-10^{-9}$ та $w_2=10^{-9}$ точні власні значення $\lambda_1=0.5-10^{-9}$, $\lambda_2=$ $=0.5+10^{-9}$, $\lambda_3=3.5$, а точні власні вектори

$$z_1=\begin{pmatrix} -\sqrt{2/15} \\ \sqrt{9/20} \\ -\sqrt{5/12} \end{pmatrix},\ z_2=\begin{pmatrix} -\sqrt{8/15} \\ \sqrt{1/20} \\ \sqrt{5/12} \end{pmatrix},\ z_3=\begin{pmatrix} \sqrt{1/3} \\ \sqrt{1/2} \\ \sqrt{1/6} \end{pmatrix}.$$

Для розв'язання задачі використано метод Якобі.

У випадку використання одинарної розрядності (real*4) у межах точності обчислень ($matheps\approx 1{,}9\times 10^{-7}$) отримано такі наближення до власних значень та наближені власні вектори:

$$\tilde{\lambda}_1=\tilde{\lambda}_2=0.5,\ \tilde{\lambda}_3=3.5,$$

$$\tilde{z}_1=\begin{pmatrix} 0.77459667 \\ -0.63245554 \\ 0 \end{pmatrix},\ \tilde{z}_2=\begin{pmatrix} 0.25819889 \\ 0.31622777 \\ -0.91287093 \end{pmatrix},\ \tilde{z}_3=\begin{pmatrix} 0.57735027 \\ 0.70710678 \\ 0.40824829 \end{pmatrix}.$$

Похибки власних значень (відповідно): 10^{-9}, 10^{-9}, 0. За результатами розв'язування перші два власні значення є кратними, тому відповідні власні вектори є наближеннями до векторів з підпростору, базисом якого є власні вектори z_1, z_2. Отже, за результатами розв'язання неможливо визначити: перші два обчислені власні значення є кратними або є близькими.

У разі використання подвійної розрядності (real*8) обчислено такі наближення до власних значень та наближені власні вектори:

$$\tilde{\lambda}_1=0.499999999,\ \tilde{\lambda}_2=0.500000001,\ \tilde{\lambda}_3=3.5,$$

$$\tilde{z}_1=\begin{pmatrix} -0.365148391830369 \\ 0.670820399515899 \\ -0.645497206451758 \end{pmatrix},\ \tilde{z}_2=\begin{pmatrix} -0.730296733168805 \\ 0.223606779250233 \\ 0.645497242284047 \end{pmatrix},$$

$$\tilde{z}_3=\begin{pmatrix} 0.577350269305096 \\ 0.707106781092267 \\ 0.408248290463863 \end{pmatrix}.$$

Тут обчислювальні похибки власних значень не перевищують $matheps \approx 2{,}2 \times 10^{-16}$. Отже, перші два власні значення не є кратними, а є близькими, а норми похибок власних векторів складають $\delta(\tilde{z}_1) = \delta(\tilde{z}_2) \approx 2.769 \times 10^{-8}$, $\delta(\tilde{z}_3) \approx 1.491 \times 10^{-10}$. Цей висновок підтверджується і для подальшого збільшення розрядності.

Для $w_1 = w_2 = 10^{-9}$ точні власні значення (кратні) $\lambda_1 = \lambda_2 = 0.5 + 10^{-9}$ та $\lambda_3 = 3.5$, а точні власні вектори $z_1 = \begin{pmatrix} -\sqrt{2/15} \\ \sqrt{9/20} \\ -\sqrt{5/12} \end{pmatrix}$, $z_2 = \begin{pmatrix} -\sqrt{8/15} \\ \sqrt{1/20} \\ \sqrt{5/12} \end{pmatrix}$, $z_3 = \begin{pmatrix} \sqrt{1/3} \\ \sqrt{1/2} \\ \sqrt{1/6} \end{pmatrix}$, причому перші два вектори (вони відповідають кратному власному значенню) визначають інваріантний підпростір. Результати на одинарній розрядності аналогічні до відповідних результатів у попередньому прикладі. На подвійній розрядності $\tilde{\lambda}_1 = \tilde{\lambda}_2 = 0.500000001$, $\tilde{\lambda}_3 = 3.5$, тобто наближені власні значення в межах точності обчислень збігаються з точними власними значеннями. Отримано такі наближення до власних векторів:

$$\tilde{z}_1 = \begin{pmatrix} 0.774596669241483 \\ -0.632455532033676 \\ 0 \end{pmatrix}, \quad \tilde{z}_2 = \begin{pmatrix} -0.258198889747161 \\ -0.316227766016838 \\ 0.912870929175277 \end{pmatrix},$$

$$\tilde{z}_3 = \begin{pmatrix} 0.577350269189626 \\ 0.707106781186547 \\ 0.408248290463863 \end{pmatrix}.$$

Тут $\tilde{z}_1 \approx -\sqrt{0.5}(z_1 + z_2)$, $\tilde{z}_2 \approx -\sqrt{0.5}(z_1 - z_2)$, а норми похибок дорівнюють 0 в межах точності обчислень. Під час збільшення розрядності два власні значення залишаються кратними. Причому цей результат не змінюється з подальшим нарощуванням розрядності.

Отже, використання багаторозрядної арифметики дає змогу визначити, чи є досліджувані власні значення кратними, чи є «патологічно» близькими.

Зауваження. В умовах наближених вихідних даних, якщо є група дуже близьких власних значень, то набагато більший сенс має обчислення довільного ортонормованого базису інваріантного підпростору, що відповідає цій групі, ніж обчислення окремих власних векторів для кожного власного значення.

ВИСНОВКИ

У роботі запропоновано адаптивні комп'ютерні методи та алгоритми для дослідження та розв'язування часткової алгебраїчної проблеми власних значень з функціями адаптивного налаштування методу, алгоритму та змінного комп'ютерного середовища на виявлені в комп'ютері математичні властивості задачі та його архітектурні особливості для забезпечення достовірності результатів розв'язування і ефективного використання обчислювальних ресурсів. Застосування створеного програмного забезпечення на основі адаптивних алгоритмів звільняє користувачів з різних предметних галузей від дослідження АПВЗ з розрідженими матрицями різної структури, ефективного розпаралелювання обчислень та даних на потрібній паралельній архітектурі, значно скорочуючи час і кошти на проведення експертних і прогнозних оцінювань у науці та інженерії.

Подальші дослідження авторів будуть зосереджені на розробленні інтелектуального інтерфейсу для математичного моделювання процесів з використанням розроблених адаптивних алгоритмів та програмних засобів.

СПИСОК ЛІТЕРАТУРИ

1. Sergienko I.V, Molchanov I.N, Khimich A.N. Intelligent technologies of high-performance computing. *Cybernetics and Systems Analysis*. 2010. Vol. 46, N 5. P. 833–844. https://doi.org/10.1007/s10559-010-9265-3.
2. Сергієнко І.В., Хіміч О.М. Математичне моделювання: Від мелм до екзафлопсів. *Вісн. НАН України.* 2019. № 8. С. 37–50. https://doi.org/10.15407/visn2019.08.037.
3. Dongarra J, Beckman P., Moore T. et al. The International Exascale Software Project roadmap. *International Journal of High Performance Computing Applications*. 2011. Vol. 25, Iss. 1. P. 3–60. https://doi.org/10.1177/1094342010391989.
4. Химич А.Н., Молчанов И.Н., Попов А.В., Чистякова Т.В., Яковлев М.Ф. Параллельные алгоритмы решения задач вычислительной математики. Киев: Наук. думка, 2008. 247 с.
5. Попов О.В., Рудич О.В., Чистяков О.В. Багаторівнева модель паралельних обчислень для задач лінійної алгебри. *Проблеми програмування*. 2018. № 2–3. С. 83–92.
6. Попов А.В., Чистяков О.В. Про ефективність алгоритмів з багаторівневим паралелізмом. *Фізико-математичне моделювання та інформаційні технології*. 2021. Вип. 33. С. 133–137. https://doi.org/10.15407/fmmit2021.33.133.
7. Чистяков О.В. Про особливості розробки програмного забезпечення для розв'язання задач на власні значення з розрідженими матрицями на гібридних комп'ютерах. *Компьютерная математика.* 2015. № 1. С. 75–84.
8. Химич А.Н., Декрет В.А., Попов А.В., Чистяков А.В. Численное исследование устойчивости композитных материалов на компьютерах гибридной архитектуры. *Проблемы управления и информатики.* 2018. № 4. С. 73–88.
9. Хіміч О.М., Чистяков О.В., Бруснікін В.М. Гібридний алгоритм узагальненого методу спряжених градієнтів для проблеми власних значень з симетричними розрідженими матрицями. *Математичні машини і системи.* 2015. № 3. С. 3–13.
10. Khimich A.N., Popov A.V., Chistyakov O.V. Hybrid algorithms for solving the algebraic eigenvalue problem with sparse matrices. *Cybernetics and Systems Analisys*. 2017. Vol. 53, N. 6. P. 937–949. https://doi.org/10.1007/s10559-017-9996-5.
11. Khimich A.N., Popov A.V., Chistyakov O.V., Sidoruk V.A. A parallel algorithm for solving a partial eigenvalue problem for block-diagonal bordered matrices. *Cybernetics and Systems Analisys*. 2020. Vol. 56, N. 6. P. 913–923. https://doi.org/10.1007/s10559-020-00311-z.
12. Molchanov I.N., Popov A.V., Khimich A.N. Algorithm to solve the partial eigenvalue problem for large profile matrices. *Cybernetics and Systems Analysis*. 1992. Vol. 28, N 2. P. 281–286. https://doi.org/10.1007/BF01126215.
13. Message Passing Interface Forum. MPI: A Message-passing Interface Standard. *International Journal of Supercomputer Applications*. 1994. Vol. 8 (3/4). P. 157–416.
14. OpenMP. Architecture Review Board. URL: http://www.openmp.org/.
15. CUDA Toolkit 11.6 Update 2 Downloads. NVIDIA Developed. URL: https://developer.nvidia.com/cuda-downloads.
16. Сидорук В.А., Єршов П.С., Богурський Д.О., Марочканич О.Р. Інтелектуалізація обчислень для задач математичного моделювання складних процесів і об'єктів. *Комп'ютерна математика*. 2019. № 1. С. 143–150.

17. Молчанов И.Н., Левченко И.С., Федонюк Н.Н., Химич А.Н., Чистякова Т.В. Численное моделирование концентрации напряжений в упругом полупространстве с двухслойным включением. *Прикладная механика.* 2002. 38. № 3. С. 65–71.

18. Lurie A.I. Theory of elasticity. Springer-Verlag, 2005. 1050 p.

19. Nikolaevskaja E.A., Chimich A.N, Chistyakova T.V. Programming with multiple precision. Ser. Studies in Computational Intelligence. Vol. 397. Berlin; Heidelberg: Springer, 2012. 234 p.

20. Khimich O.M., Chistyakova T.V, Sidoruk V.A., Yershov P.S. Adaptive computer technologies for solving problems of computational and applied mathematics. *Cybernetics and Systems Analysis.* 2021. Vol. 57, N 6. P. 990–997. https://doi.org/10.1007/s10559-021-00424-z.

21. Buttari A., Langou J., Kurzak J., Dongarra J. A class of parallel tiled linear algebra algorithms for multicore architectures. *Parallel Computing*. 2009. Vol. 35, Iss. 1. P. 38–53. https://doi.org/10.1016/j.parco.2008.10.002.

22. Pissanetzky S. Sparse matrix technology. 1st ed. Elsevier, 1984. 312 p.

23. Saad Y. Iterative methods for sparse linear systems. 2nd edition. SIAM, 2003. 547 p.

24. Попов А.В. Параллельные алгоритмы решения линейных систем с разреженными симметричными матрицами. *Проблеми програмування.* 2008. № 2–3. С. 111–118.

25. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. Москва: Мир, 1984. 334 с.

26. Parlet B.N. The symmetric eigenvalue problem. SIAM, 1998. 391 p. https://doi.org/10.1137/1.9781611971163.

27. Попов О.В., Рудич О.В. До розв'язування систем лінійних рівнянь на комп'ютерах гібридної архітектури. *Математичне та комп'ютерне моделювання. Сер. Фізико-математичні науки.* 2017. Вип. 15. С. 158–164.

28. Суперкомп'ютерний комплекс СКІТ. URL: http://icybcluster.org.ua.

29. The SuiteSparse Matrix Collection. URL: https://cise.ufl.edu/research/sparse/matrices.

O.M. Khimich, O.V. Popov, O.V. Chistyakov, V.O. Kokhanovskyi

ADAPTIVE ALGORITHMS FOR SOLVING EIGENVALUE PROBLEMS IN A VARIABLE COMPUTER ENVIRONMENT OF SUPERCOMPUTERS

Abstract. The authors propose software for the analysis and solution of the algebraic eigenvalue problem using a MIMD computer with GPUs, which includes parallel algorithms and programs with the functions of adaptive configuration of the variable computer environment (multilevel parallelism, variable topology of interprocessor communications, mixed word length, caching, etc.) on the mathematical properties of the problem identified in the computer and the architectural features of the computer to ensure the reliability of the solution results with the efficient use of computing resources.

Keywords: algebraic eigenvalue problem (APEV), changeable computing environment, adaptive algorithms, gradient conjugation method, subspace iteration method, multi-core computers of MIMD-architecture with graphic processors**.**

Надійшла до редакції 23.01.2023