НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ

ІНСТИТУТ КІБЕРНЕТИКИ ІМЕНІ В.М. ГЛУШКОВА

Кваліфікаційна наукова
праця на правах рукопису

Єршов Павло Сергійович

УДК 519.8

ДИСЕРТАЦІЯ

ІНТЕЛЕКТУАЛЬНА СИСТЕМА КОМП'ЮТЕРНОЇ МАТЕМАТИКИ ДЛЯ МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ В НАУЦІ І ІНЖЕНЕРІЇ

113 – «Прикладна математика»

Галузь знань 11 – «Математика та статистика»

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Єршов П.С.

Науковий керівник: Хіміч Олександр Миколайович
доктор фізико-математичних наук,
професор, академік НАН України

Київ – 2023

АНОТАЦІЯ

Єршов П.С. Інтелектуальна система комп’ютерної математики для математичного моделювання в науці і інженерії. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 113 “Прикладна математика”. – Інститут кібернетики імені В.М. Глушкова Національної академії наук України, Київ. – 2023.

Зміст дисертації. У **вступі** обґрунтовано актуальність теми, сформульовано мету та задачі досліджень, розкрито наукову новизну та практичну цінність роботи, представлено її загальну характеристику.

Перший розділ присвячений огляду літератури за темою дисертації та обґрунтуванню вибору напрямку досліджень.

У **другому розділі** досліджено математичний апарат комп’ютерного дослідження лінійних систем з наближеними даними; розроблено паралельний блочний алгоритм LL^T-розвинення «хмарочосної» матриці, досліджено прискорення та ефективність паралельного алгоритму.

У **третьому** розділі розроблено алгоритми автоматичного визначення структури розріджених матриць та їх регуляризації на основі методів машинного начання та нейромережевих технологій. Побудовано згортковану нейронну мережу «Sparse Matrix Vision». Виконана програмна реалізація та проведено чисельні експерименти.

У **четвертому** розділі розроблено принципи, архітектуру та склад інтелектуальної системи комп’ютерної математики (ІСКМ) для автоматизації процесу дослідження та розв’язування задач, запропоновано її комп’ютерну

реалізацію. Наведено результати практичних розрахунків. Досліджено ефективність алгоритмічного і програмного забезпечення ІСКМ на низці задач математичного моделювання в будівельній галузі.

Ключові слова: система комп’ютерної математики, паралельні алгоритми, розріджені матриці, комп’ютери гібридної архітектури, машинне навчання, нейронні мережі.

ABSTRACT

Yershov P.S. Intelligent system of computer mathematics for mathematical modeling in science and engineering. – Qualifying scientific work on manuscript rights.

Dissertation for a Doctor of Philosophy Degree by specialty 113 Applied mathematics. – V.M. Glushkov Institute of Cybernetics of the National Academy of Science of Ukraine. – Kyiv, 2023.

The contents of the dissertation. The **introduction** substantiates the relevance of the topic, formulates the purpose and tasks of the research, reveals the scientific novelty and practical value of the work, and presents its general characteristics.

The **first chapter** is devoted to the review of the literature on the topic of the dissertation and justification of the choice of research direction.

The **second chapter** explores the mathematical apparatus of computer research of linear systems with approximate data; a parallel block algorithm for LL^T-factorization of "cloud" matrix was developed, the acceleration and efficiency of parallel algorithms were investigated.

In the **third chapter**, algorithms for automatic identification of the structure of sparse matrices and their regularization based on machine learning methods and neural network technologies are developed. A convolutional neural network "Sparse Matrix Vision" was built. Software implementation and numerical experiments were carried out.

In the **fourth chapter**, the principles, architecture, and composition of the intelligent system of computer mathematics (ISCM) for automating the process of research and problem solving are developed, and its computer implementation is proposed. The results of practical computations are presented. The effectiveness of

algorithmic and software ISCM on a number of problems of mathematical modeling in the construction industry was studied.

Key words: computer mathematics, linear algebra, sparse matrices, hybrid architecture computers, artificial neural networks

Список публікацій здобувача

Публікації, в яких опубліковано основні наукові результати дисертації

1. 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, №6. P. 990–997. DOI: https://doi.org/10.1007/s10559-021-00424-z SCOPUS (Q2)
2. Хіміч О.М., Чистякова Т.В., Сидорук В.А., Єршов П.С. Адаптивні алгоритми дослідження задач в змінному комп'ютерному середовищі. *Фізико-математичне моделювання та інформаційні технології*. 2021. Вип. 33. С. 181-185.
   DOI: https://doi.org/10.15407/fmmit2021.33
3. Чистякова Т.В., Єршов П.С. Про вибір розрядності обчислень в інтелектуальній системі обробки матриць. *Математичне та комп'ютерне моделювання. Серія: Фізико-математичні науки*. 2019. Вип. 19. С. 193–198.
   DOI: https://doi.org/10.32626/2308-5878.2019-19.193-198
4. Сидорук В.А., Єршов П.С. Адаптивний алгоритм розв'язання систем рівнянь з блочно-хмарочосними матрицями. *Міжнародний науково-технічний журнал «Проблеми керування та інформатики»*. 2022. №5. С. 17–31.
   DOI: https://doi.org/10.34229/2786-6505-2022-5-2
5. Хіміч О.М., Чистякова Т.В., Сидорук В.А., Єршов П.С. Інтелектуальна система комп'ютерної математики INPARSOLVER. *Штучний інтелект*. 2020. № 4, т.25. С. 60–71. DOI: https://doi.org/10.15407/jai2020.04.060
6. Чистяков О.В., Ніколайчук О.О., Єршов П.С. Про математичне моделювання задач стійкості конструкцій на сучасних комп'ютерах. *Компьютерная математика*. 2018. № 2. С. 30−37.

7. Сидорук В.А., Єршов П.С., Богурський Д.О., Марочканич О.Р. Інтелектуалізація обчислень для задач математичного моделювання складних процесів і об’єктів. *Компьютерная математика*. 2019. № 1. С. 143–150.

Публікації, що засвідчують апробацію матеріалів дисертації

8. Khimich A., Sydoruk V., Yershov P. Intellectualization Of Computation Based On Neural Networks For Mathematical Modeling. *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, 18-20 Dec. 2019. IEEE: 2019. P.445-448. DOI: https://doi.org/10.1109/ATIT49449.2019.9030444

ЗМІСТ

ВСТУП

Високопродуктивні обчислення на сьогодні є одним з основних засобів наукових та інженерних досліджень. Постійно виникають нові нагальні задачі математичного моделювання з різних предметних областей, які потребують значних обчислювальних ресурсів. Зростання потужностей сучасних суперкомп'ютерів здійснюється за рахунок ускладнення архітектури багатопроцесорних обчислювальних систем з розподіленою та ієрархічною спільною пам'яттю, зі складною організацією багатопотокових і векторизованих операцій, із застосуванням різного типу прискорювачів. Наразі у математичному моделюванні широко використовуються комп'ютери гібридної архітектури, яка поєднує MIMD-архітектуру багатоядерних комп'ютерів (CPU) з SIMD-архітектурою графічних прискорювачів (GPU).

Разом із зростанням потужностей комп'ютерів ростуть і проблеми їх експлуатації. Збільшення кількості процесорів у паралельних комп'ютерах або використання GPU в гібридних системах означає істотне збільшення комунікаційних витрат і зниження їх ефективності. Як показують теоретичні та практичні дослідження, вже зараз за рахунок комунікаційних витрат наявна істотна різниця між максимальною продуктивністю і експлуатаційною [1–4].

Важливо відмітити, що за останні роки темпи розвитку комп'ютеризації не знижуються, про що свідчить список Топ-500 найпотужніших суперкомп'ютерів світу за 2023 рік [5]. В них використовуються різні апаратні складові: процесори (Intel, IBM, AMD, HP, NEC, Cray), співпроцесори, мережеві карти (Ethernet, Myrinet, Infiniband, SCI). Вони функціонують під управлінням різних

операційних систем (Unix, Linux, Windows) та реалізують різне прикладне програмне забезпечення при використанні різних мов програмування. Застосування такої різноманітної комп'ютерної техніки у прикладних дослідженнях для користувачів ще більше ускладнюється.

Проблеми достовірності розв'язків із зростанням обсягів задач, що розв'язуються на таких комп'ютерах, також ускладнюються. Це відбувається з багатьох причин, але перш за все через похибки вихідних даних задач, відмінності властивостей математичних і комп'ютерних моделей, особливостей машинної арифметики тощо [1–2, 4, 6–10].

Актуальність теми. Математичні моделі фізичних процесів, які описуються диференціальними та інтегральними рівняннями або варіаційними задачами завжди мають наближені дані, що обумовлені похибками вимірювань, спостережень, припущень, гіпотез і т. п. У цьому випадку вихідні дані є деяким наближенням до точних даних задачі з деякими похибками. Надалі при дискретизації математичної моделі ці похибки трансформуються в похибки коефіцієнтів рівнянь, які підлягають розв'язуванню. Крім того, вихідні дані математичних моделей можуть бути задані точно в числовому вигляді або представлені математичними формулами, але заокруглення при введенні чисел в комп'ютер та обчислення формул також призводять до комп'ютерних моделей з наближеними даними.

Математичне моделювання значної частини процесів та явищ зводиться до розв'язування систем лінійних алгебраїчних рівнянь (СЛАР) з розрідженими матрицями різної структури та наближеними даними великих розмірів. Від достовірного та ефективного розв'язування на комп'ютері саме цих задач у

значній мірі залежить ефективність математичного моделювання різних фізико-механічних процесів. Але через наближений характер даних математичні властивості комп'ютерної та вихідної задачі можуть відрізнятися, до прикладу, невироджена матриця може стати виродженою і навпаки. Для забезпечення достовірності комп'ютерних розв'язків необхідно розробити технологію дослідження в комп'ютерному середовищі математичних властивостей комп'ютерної моделі, зокрема з використанням багаторозрядної арифметики [10–12].

Крім того, актуальною проблемою є розроблення комп'ютерних методів визначення в комп'ютері структури матриць надвеликих порядків, що виникають у практичних задачах, та застосування ефективних способів їх обробки. Ефективне зберігання та використання ненульових елементів розріджених матриць в комп'ютері дають можливість значно зменшити обчислювальні ресурси. В залежності від поставленої задачі та особливостей виду матриці, можна оптимізувати її портрет шляхом зміни ширини стрічки, зменшення профілю матриці, корегування загального заповнення, зведення структури до регулярного виду тощо [10].

Самостійне вирішення вказаних проблем потребує від користувачів – спеціалістів із різних предметних областей відповідних знань в області обчислювальної математики, архітектурних особливостей комп'ютерів, навиків використання різних технологій паралельного програмування і т. д.

Отже, необхідно розробляти програмне забезпечення рівня кінцевого користувача – інтелектуальну систему комп'ютерної математики (ІСКМ), що забезпечуватиме автоматичне адаптивне налаштування методу, алгоритму та

топології комп’ютера на математичні властивості комп’ютерної задачі з наближеними даними, в тому числі для розріджених матриць великих порядків та отримання розв’язку з оцінкою достовірності результатів [11–14].

Застосування ІСКМ надасть можливість суттєво перерозподілити роботи по постановці і розв’язанню задач між користувачем і комп’ютером у порівнянні з традиційними технологіями, скоротити терміни розроблення прикладних додатків для розв’язання науково-технічних задач і підвищити якість одержуваних комп’ютерних розв’язків. Таким чином, ІСКМ надасть можливість розв’язувати нові науково-технічні задачі, істотно скоротити кошти і час на розробку об’єктів сучасної техніки, що і визначає актуальність завдань дисертаційної роботи.

Мета й завдання дослідження. Мета дисертаційної роботи розробити принципи, алгритмічне та програмне забезпечення інтелектуальної системи комп’ютерної математики для автоматизації процесу дослідження та розв’язування систем лінійних алгебраїчних з розрідженими матрицями на комп’ютерах гібридної архітектури.

Для досягнення мети дослідження поставлено такі завдання:

- розробити принципи, архітектуру та склад ІСКМ для автоматизації процесу дослідження та розв’язування СЛАР з розрідженими матрицями на комп’ютерах гібридної архітектури;
- розробити та обґрунтувати методи та адаптивні комп’ютерні алгоритми для дослідження та розв’язування СЛАР з матрицями розрідженої структури для гібридних комп’ютерів;
- розробити алгоритми дослідження в комп’ютерному середовищі математичних властивостей СЛАР з наближеними даними, в тому числі

принципів використання багаторозрядної арифметики для забезпечення достовірності розв’язків;

- розробити методи розпізнавання структури розрідженої матриці на основі машинного навчання та нейромережевих технологій для вибору ефективного алгоритму розв’язування ;
- реалізувати ІСКМ для розв’язування СЛАР з наближеними даними на комп’ютерах гібридної архітектури;
- провести експериментальне дослідження функціональних можливостей розробленої ІСКМ на комп’ютерах гібридної архітектури (до прикладу, суперкомп’ютерах родини кластерів СКІТ) на розв’язані практичних задачах;
- інтегрувати ІСКМ у математичне забезпечення суперкомп’ютера родини кластерів СКІТ.

Об’єкт дослідження – інтелектуальна система комп’ютерної математики для математичного моделювання в науці та інженерії (дослідження та розв’язування СЛАР) на комп’ютерах гібридної архітектури.

Предмет дослідження – системи лінійних алгебраїчних рівнянь з розрідженими матрицями та наближеними даними, паралельні алгоритми дослідження та розв’язування; методи штучного інтелекту для комп’ютерного розпізнання структур розріджених матриць, оцінки достовірності результатів з використанням багаторозрядної арифметики.

Методи дослідження: чисельні методи лінійної алгебри, методи паралельних обчислень, теорія похибок та методи штучного інтелекту.

Наукова новизна отриманих результатів. Наукову новизну роботи визначають принципи, алгритмічне та програмне забезпечення ІСКМ для автоматизації процесу дослідження та розв’язування СЛАР з розрідженими матрицями на комп’ютерах гібридної архітектури, а саме:

– *вперше* запропоновано принципи автоматизації процесу дослідження та розув’язуваання СЛАР з розрідженою структурою даних;

– *вперше* розроблено паралельний алгоритм трикутної факторизації матриць блочно-хмарочосної структури;

– *вперше* розроблено метод розпізнавання структури розрідженої матриці на основі машинного навчання та згорткових нейронних мереж для вибору ефективного алгоритму розв’язування;

– *вперше* запропоновано комп’ютерний алгоритм ідентифікації математичних властивостей матриць (виродженість, погана обумовленість) з використанням багаторозрядної арифметики для забезпечення достовірності комп’ютерних розв’язків;

– створено інтелектуальну систему комп’ютерної математики для автоматизації дослідження та розв’язування СЛАР з розрідженими матрицями з функцією адаптивного налаштування методу, алгоритму, топології паралельного комп’ютера на математичні властивості задачі.

Практичне значення отриманих результатів. Отримані в дисертації результати мають теоретичне та практичне значення для моделювання фізико-механічних процесів, що зводяться до розв’язування СЛАР на сучасних високопродуктивних суперкомп’ютерах. Створена ІСКМ може використовуватися в галузях, які мають на сьогодні дуже важливе значення при створенні нових композитних матеріалів в літакобудуванні, суднобудуванні, ракетобудуванні, дослідженнях міцності конструкцій промислового і цивільного

будівництва, зварних конструкцій. Застосування ІСКМ надасть можливість суттєво перерозподілити роботи по постановці і розв'язуванню задач між користувачем і комп'ютером у порівнянні з традиційними технологіями, скоротити терміни розроблення прикладних застосунків для розв'язання науково-технічних задач і підвищити якість одержуваних комп'ютерних розв'язків. Наразі розв'язано низку тестових і практичних задач міцності і стійкості конструкцій.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконана у відповідності до планів наукових досліджень відділу № 150 Інституту кібернетики імені В.М. Глушкова НАН України в рамках науково-дослідницьких тем:

ВФ.150.26 «Розробити моделі та методи гетерогенних обчислень для задач механіки суцільних середовищ» (№ держреєстрації 0119U002224, 2019–2023 рр.);

ВП.150.4.1230 «Розробити платформу високопродуктивних обчислень на базі суперкомп'ютера СКІТ для задач кібербезпеки, математичного моделювання, інженерії» (№ держреєстрації 0123U101573, 2023 рр.);

ВП.150.30 «Розробити інтелектуальний інтерфейс для моделювання фізико-механічних процесів на основі паралельних обчислень» (№ держреєстрації 0122U000906, 2022–2024 рр.).

Особистий внесок здобувача. Автором самостійно отримано основні результати дисертаційного дослідження. В опублікованих у співавторстві наукових працях здобувачем здійснено: у статті [84] – розроблення методів машинного навчання на основі нейронних мереж для комп'ютерного визначення

структури розріджених матриць великих порядків; у статті [91] – реалізація факторизації стрічкової симетричної додатно означеної матриці паралельним блочним алгоритмом LL^{T}–розвинення для персонального комп'ютера з новітніми процесорами Intel Xeon Phi; у статті [70] – розроблення ІСКМ для достовірного розв'язування СЛАР на багатоядерному комп'ютері з графічними процесорами, яка здатна автоматично досліджувати математичні властивості комп'ютерних моделей задач та приймати рішення щодо ефективного їх розв'язування; у статті [71] – розроблення бази знань для автоматичного дослідження та розв'язування СЛАР з матрицями розрідженої структури, реалізація InparSolver на гібридному комп'ютері СКІТ-4 кластерного комплекса СКІТ; у публікації [85] – розроблення технології автоматичної класифікації розріджених матриць з апріорно невизначеною структурою (на основі штучних нейронних мереж) та вибору ефективного відповідного алгоритму розв'язування СЛАР; у статті [88] – розроблення адаптивних алгоритмів дослідження структурних і математичних властивостей СЛАР з використанням багаторозрядної арифметики і нейромережевих технологій; у статті [72] – концепція створення ІСКМ для розріджених матриць, в основі якої лежить триада: комп'ютерна математика, високопродуктивні обчислення, інтелектуальне програмне забезпечення. Застосування цих принципів організації обчислень дає можливість істотно перерозподілити роботи по постановці і розв'язуванню задач між користувачем і комп'ютером у порівнянні з традиційними технологіями. В статті [89] – розроблення та дослідження нового адаптивного паралельного алгоритму з розрідженими симетричними матрицями блочно-хмарочосного виду , який враховує структуру розріджених матриць та

їхню наповненість даними, отримання оцінок прискорення алгоритму на різній кількості процесорних ядер із застосуванням різної величини блоків, використаних для реалізації обчислень.

Апробація результатів дисертації. Результати дисертації доповідались та обговорювались на:

- XIX Міжнародна науково-технічна конференція «Штучний інтелект та інтелектуальні системи» 15 – 18 жовтня 2019 р., м. Київ, Україна;
- Міжнародний науковий симпозіум «Питання оптимізації обчислень (ПОО-XLVI)», 22–24 вересня 2019 р., м. Кам'янець-Подільський, Україна;
- XX Міжнародна науково-технічна конференція «Штучний інтелект та інтелектуальні системи (AIIS'2020), 27 листопада, 2020, м. Київ, Україна;
- Міжнародна наукова конференція присвячена 30-річчю незалежності України «Питання оптимізації обчислень» (ПОО-XLVIII) 21–24 вересня, 2021, м. Львів, Україна.

Публікації. Наукові результати дисертаційної роботи у повній мірі викладено у 8 публікаціях, з яких: одна стаття у періодичному науковому виданні, проіндексованому у базі Scopus (віднесеному до другого квартиля Q2), 6 статтей у наукових виданнях, включених до переліку наукових фахових видань України, одні тези доповіді на міжнародній конференції опубліковано англійською мовою та проіндексовано у базі Scopus.

Структура та обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, загальних висновків, списку використаних літературних джерел, який містить 104 найменування. Загальний обсяг дисертаційних досліджень викладено на 124 сторінках друкованого тексту, де

обсяг основного тексту – 104 сторінки. Дисертація включає 16 рисунків, 3 таблиці.

РОЗДІЛ 1

ОГЛЯД ЛІТЕРАТУРИ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1.1 Постановка та дослідження математичних моделей з наближеними даними

Характерною особливістю математичних моделей з наближеними вихідними даними є те, що їх математичні властивості апріорі невідомі. Машинна модель задачі, яку в результаті й доводиться розв’язувати на комп’ютері, має завжди наближений щодо вихідної задачі характер або через похибку вихідних даних, або через похибку отримання (вводу) числових даних про задачу в комп’ютері. Тому властивості машинної моделі задачі можуть відрізнятися від властивостей математичної задачі. Ще однією важливою рисою чисельного моделювання є наближений характер машинних обчислень, обумовлений обмеженою довжиною машинного слова комп’ютерів [10].

Ще на початку 70-х років було помічено, що при використанні відомих високоефективних бібліотек програм чисельного аналізу, таких як NAG, IMSL, Bell Library, Boeining Library та інших, в деяких випадках отримуються комп’ютерні результати, які не містять фізичного змісту. Для дослідження таких проблем було створено програму NATS (National Aktivity to Test Software), за результатами яких було прийнято рішення створювати пакети програм для розв’язування задач обчислювальної математики бібліотечного типу, які не залежать від математичних і технічних особливостей комп’ютерів. Тобто бібліотеки програм складаються з програмних модулів, що реалізують логічно

закінчені частини алгоритмів, з яких користувач може самостійно створювати програму розв’язування задачі та аналізувати достовірність отримуваних комп’ютерних результатів. Прикладом таких програмних засобів є відомі пакети програм LINPACK [15] для розв’язування СЛАР та EISPACK для розв’язування алгебраїчної проблеми власних значень [16] на однопроцесорних комп’ютерах.

Проблемам достовірності комп’ютерних результатів розв’язування таких задач присвячені наукові роботи багатьох відомих у світі вчених, до прикладу, Дж. Х. Уілкінсона , К. Райнша, Дж. Донгарри, В.В. Воєводіна, Д.К. Фадеева, В.Н. Фадеєвої, А.А. Дородниціна та інших. В Інституті кібернетики ім. В.М. Глушкова НАН України на протязі багатьох років проводяться наукові дослідження достовірності комп’ютерних розв’язків задач математичного моделювання процесів з різних галузей авторами: І.В. Сергієнко, О.М. Хіміч, В.К. Задірака, О.В. Попов, І.М. Молчанов та інші.

Одним з найпростіших шляхів мінімізації помилок, пов’язаних з округленнями і втратою точності при виконанні обчислень, є подальше зростання розрядності обчислень. Вирішенням цих питань на апаратному рівні розробники комп’ютерів почали займатися ще при розробці перших комп’ютерів. Так, до прикладу, довільна розрядність обчислень з’явилася на комп’ютерах серії МІР ще в 1965 році.

Практичні задачі, які доводиться розв’язувати на суперкомп’ютерах різної архітектури, особливо чутливі до точності комп’ютерних обчислень. Як показали результати наукових досліджень [17–22], для забезпечення достатньої точності результатів розв’язування деяких практичних задач необхідно використовувати дворазове підвищення розрядності, для інших – чотириразове,

а є також задачі, що потребують сотні розрядів обчислень. У зв'язку з цим все більшої актуальності набуває використання багаторозрядної арифметики.

З метою розв'язання проблем, які пов'язані з точністю отриманих комп'ютерних результатів та стандартизацією компіляції програм на комп'ютерах різної архітектури, у серпні 2008 року був опублікований стандарт IEEE 754-2008 [23], який замінив раніше діючий стандарт обчислень з плаваючою комою IEEE 754-1985.

Стандарт 1985 передбачав 2 типи чисел з плаваючою комою: одинарну (32-розрядну) і подвійну (64-розрядну) точності. Однією з перших його апаратних реалізацій став арифметичний співпроцесор Intel 8087, в якому додатково в якості внутрішнього формату використовувався «розширений» 80-розрядний формат з 64-розрядною мантисою і 16-розрядним порядком.

Основним нововведенням в стандарті 2008 року став 128-розрядний формат «quad-double», тобто формат «почетверенної точності», поле мантиси в якому розширено до 113 розрядів. Проте апаратна підтримка цього формату при розробці комп'ютерів потребує значних затрат і на даний час не вирішена [24].

Більш поширеними способами підвищення точності комп'ютерних результатів є на сьогодні багаторозрядна комп'ютерна арифметика – програмна обробка чисел, розрядність яких перевищує стандартну довжину машинного слова в комп'ютері (double).

Розглянемо алгоритмічно-програмні засоби, що підтримують комп'ютерні обчислення з підвищеною розрядністю:

1) Формат типу даних long-double, який можна реалізовувати в програмах, написаних на мові C++ (типовий розмір: 8, 12 або16 байт).

2) Формати типів даних double-double, quad-double. Це альтернативні підходи, що полягають у багатокомпонентному форматі зберігання та використання чисел. Число виражається у вигляді необчисленої суми звичайних чисел із плаваючою комою, кожне з яких – саме значення та показник. Такі формати реалізовано, до прикладу, у бібліотеці форматів QD [25]. За таким підходом можна представляти набагато більший діапазон чисел, тоді як попередній багатокомпонентний підхід має перевагу в швидкості. Отже, double-double – це необчислена сума двох чисел подвійної точності IEEE стандарту, яка може представляти принаймні 13 байт (106 біт) значущих знаків. Подібним чином число quad-double – це невизначена сума чотирьох чисел подвійної точності IEEE-стандарту, здатна представляти близько 26 байт (212 біт) значущих знаків. Функції бібліотеки QD для реалізації таких типів форматів можна використовувати у C++ програмах.

Бібліотеки програм, за допомогою яких реалізується робота з числами на підвищеній розрядності:

- ARPREC [26] – пакет програм довільної точності включає процедури арифметичних обчислень, а також багато алгебраїчних і трансцендентних функцій. Підтримує обчислення з дійсними, цілими та комплексними числами. Має інтерфейси C++ та Фортран-90.
- GMP (GNU Multiple-Precision Library) [27] – бібліотека, що має великий набір оптимізованих процедур для підтримки обчислень з цілими, раціональними та дійсними числами з використанням довільної розрядності. Перша версія появилася в 1991 році та постійно розвивається. Може використовуватися на різних операційних

платформах – Unix, GNU / Linux, Solaris, HP-UX, Windows. Має розвинутий інтерфейс C++, може використовувати в Ci. Можна змінювати розрядність в одному і тому ж програмному модулі в різних місцях реалізації алгоритму. Режими округлення, сумісні зі специфікаціями IEEE-754, не підтримуються.

- MPFR [28] – розширення GMP, що забезпечує обчислення довільної точності з можливістю використання одного з чотирьох режимів округлення, що відповідають стандарту IEEE-754. Точність може бути встановлена окремо для кожної змінної. Нормалізовані числа не підтримуються. Має більш високу швидкодію у порівнянні з багатьма аналогами.

3) Системи комп'ютерної математики, в яких реалізовано обчислення з підвищеною розрядністю: Mathematica [29], MATLAB [30], Maple [31] за рахунок символьного процесора MathCAD [32].

1.2 Ефективне використання матриць різної структури

Математичне моделювання процесів, які виникають в аналізі складних багатокомпонентних середовищ, показало, що вони досить часто зводяться до розв'язування СЛАР з матрицями надвеликих порядків різної структури (щільних, стрічкових, розріджених довільної структури).

Багато методів розв'язування задач лінійної алгебри базуються на розвиненні матриці в добуток матриць стандартних виглядів, до прикладу, нижньої і верхньої трикутних матриць. Алгоритми таких розвинень

характеризуються поступовим зменшенням від кроку до кроку розміру оброблюваної частини матриці. Тому важливо забезпечити приблизно однакові об’єми обчислень, обмінів та синхронізацій, що виконуються кожним процесом або потоком на паралельній моделі обчислень, тобто виключити вплив ефекту Гайдна [10]. Як показали дослідження, достатньо хорошу збалансованість завантаження процесів (потоків) забезпечують паралельні алгоритми, в яких використовуються так звані циклічні схеми розподілу і обробки матриць [10].

Найчастіше прикладні задачі у математичному моделюванні фізико-механічних процесів зводяться до розв’язування СЛАР з розрідженими матрицями. Структура ненульових елементів (структура розрідженої матриці), що визначається нумерацією невідомих задачі, може бути як регулярною (до прикладу, стрічковою, профільною, блочно-діагональною з обрамленням тощо), так і нерегулярною (довільною) [10, 40–44].

Існує ціла низка методів, які дозволяють регулювати заповнення матриці під час реалізації розв’язування задач на комп’ютерах. Розроблено як методи загального призначення [40, 44], так і ті, що орієнтовані на матриці конкретного виду [41, 50]. В залежності від поставленої задачі та структурних особливостей матриці, можна оптимізувати її портрет шляхом зміни ширини стрічки, зменшення профілю матриці, корегування загальної кількості заповнення, зведення структури до певного регулярного виду.

Схеми зберігання розріджених матриць. Існують різні схеми зберігання, які відрізняються способом використання нулів. У деяких випадках допускається зберігання частини нулів в обмін на спрощення схеми зберігання; в інших − використовуються всі нулі матриці; в третіх − нулі взагалі не використовуються.

Вибір формату зберігання, природно, впливає на запити до пам’яті, а отже істотно впливає на ефективність програмних реалізацій алгоритмів обробки розріджених матриць. Найчастіше використовуються такі формати зберігання елементів розріджених матриць: координатний формат, розріджений рядковий або стовпчиковий формат, формат ELLPaCK, гібридний формат тощо [10].

1.3 Застосування методів та алгоритмічно-програмного забезпечення для розв’язування лінійних систем

Інтерес до проблеми побудови ефективних методів розв’язування СЛАР з матрицями різної структури зумовлений головним чином їх численними застосуваннями. Зокрема, системи рівнянь з розрідженими матрицями з’являються в задачах аналізу міцності конструкцій у цивільному та промисловому будівництві, в авіабудуванні, ракетобудуванні, суднобудуванні тощо. На сьогодні області застосування методів розв’язування СЛАР з розрідженими матрицями постійно розширюються

Методи розв’язування СЛАР та бібліотеки програм на їх основі. Методи розв’язування СЛАР і пов’язаних з ними задач найбільш повно викладені у роботах [10, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 48, 49, 50]. Розвиток методів розв’язування СЛАР з розрідженими матрицями представлено у монографіях С. Писанецьки, Й. Саада, А. Джорджа, Дж. Лю, О. Естербю, М. Златова, Р. Тюарсона, Дж. Голуба та інших.

В даний час паралельні комп’ютери різної архітектури стали основними обчислювальними засобами для чисельного моделювання складних процесів у

різних дисциплінах. Такі комп'ютери дають можливість розв'язувати задачі більших обсягів, часто за менший час. Значна їх частина зводиться до розв'язування СЛАР з розрідженими матрицями. Тому розроблення паралельних алгоритмів для розв'язування СЛАР з розрідженими матрицями великих порядків на сучасних високопродуктивних суперкомп'ютерах є дуже актуальною темою. Розробці паралельних алгоритмів для розв'язування систем з розрідженими матрицями присвячені роботи Дж. Донгари, Дж. Голуба, Г. Алана, Дж.М. Ортеги, О.М. Хіміча, І. М. Молчанова, Т.Л. Фрімана, Дж. Квіна та інших.

Створені методи для розв'язування задач лінійної алгебри стали основою розробки алгоритмів і програм для розв'язування СЛАР для різних архітектур комп'ютерів, з різними структурами даних. За останні кілька десятиліть на основі розроблених алгоритмів було створено ряд програмних бібліотек, включаючи програми для розв'язування СЛАР з розрідженими матрицями: Bell Laboratories, SparseBLAS, NAG, Boeing Library, SparsPak, SSP, IMSL тощо. Серед програмних засобів, розроблених для паралельних комп'ютерів, бібліотеки MUMPS, PSparslib, Aztec, BlockSolver95, SPARSKIT, Hypre, ILUS, PSBLAS, PARMS, PSPASES, SUPERLU забезпечують ефективну реалізацію алгоритмів з розрідженими матрицями.

Окремо слід відзначити алгоритмічно-програмне забезпечення з лінійної алгебри створене під керівництвом проф. J. Донгари. Широко відомі пакети програм, що реалізують блочні алгоритми розв'язування СЛАР: LAPACK – для однопроцесорних комп'ютерів, ScaLAPACK – для MIMD-комп'ютерів, а також бібліотека програм матрично-матричних та матрично-векторних операцій BLAS.

Блочні алгоритми дають можливість узгоджувати розмір блоку матриці системи з розміром кеш-пам’яті комп’ютера, що значно підвищує швидкодію алгоритмів та програм. На основі програмних модулів з цих пакетів розроблено бібліотеки програм для комп’ютерів різної архітектури.

Під керівництвом проф. Донгарри розроблено серію бібліотек програм з лінійної алгебри для гібридних комп’ютерів. На основі бібліотеки BLAS створено бібліотеку CuBLAS для платформи NVIDIA®CUDA™, а для розріджених матриць – аналогічну бібліотеку CuSparse. Для гібридних комп’ютерів слід виділити бібліотеку програм MAGMA, орієнтовану на роботу із щільними матрицями на гібридних архітектурах (CPU + GPU, CPU + Xeon Phi). Для реалізації автоматичного планування обчислень на CPU і GPU у бібліотеці програм MAGMA використовується система StarPU, набір планувальників якої надають можливість задати ефективну модель обчислень на CPU та GPU, динамічно планувати обчислення з урахуванням витрат часу виконання кожної підзадачі на кожному пристрої, вибрати оптимальний розмір блоку матриці на GPU для максимального його заповнення і т. д.

Також на основі розглянутих програмних продуктів з лінійної алгебри створено відповідні бібліотеки програм, адаптовані до конкретних процесорів: ACML (Core Math Library AMD) – оптимізована під процесори компанії AMD, Intel MKL (Intel Math Kernel Library) – оптимізована під процесори компанії Intel.

Бібліотека IntelMKL підтримує багатопоточне розпаралелювання на багатоядерних комп’ютерних системах. Однією з особливостей Intel MKL є незалежність від компіляторів. Це означає, що код програми, написаний для

однієї комп’ютерної системи, може бути вільно перенесений на інші системи. Крім того, компоненти бібліотеки розділені по незалежним рівням. Інша особливість Intel MKL полягає в тому, що розробнику програм розв’язування задач не потрібно турбуватися про налаштування на конкретний процесор. Бібліотека Intel MKL здійснює диспетчеризацію програмного коду таким чином, що на стадії виконання на кожному типі процесора запускаються оптимальні для нього процедури з урахуванням обсягу задачі.

Ефективною є також бібліотека програм CUSP, розроблена співробітниками фірми NVIDIA Research Джаредом Хобероком (Jared Hoberock) і Натаном Беллом (Nathan Bell), яка реалізує розв’язування розріджених лінійних систем ітераційними паралельними алгоритмами на основі методу спряжених градієнтів з різними передообумовлювачами.

Для розв’язування ітераційними методами лінійних систем та задач на власні значення, які виникають при чисельному розв’язуванні рівнянь з частковими похідними, японськими спеціалістами створено паралельну бібліотеку програм Lis (Library of Iterative Solvers).

Як показує огляд літератури та досвід використання розглянутих вище високопродуктивних бібліотек програм з обчислювальної математики, в них передбачається неявно, що початкові дані задач задано точно і аналіз достовірності комп’ютерних результатів не проводиться.

1.4 Особливості створення адаптивних методів та алгоритмів розв’язування СЛАР для гібридних комп’ютерів

Аналізуючи проблеми ефективного використання високопродуктивних комп’ютерів, архітектурні та технологічні особливості яких постійно змінюються, приходимо до створення адаптивних алгоритмів розв’язування задач, які можуть автоматично налаштовувати метод, комп’ютерний алгоритм, необхідну модель розпаралелення так ефективне (змінне) комп’ютерне середовище на конкретну задачу [1–4, 6–9].

На сучасних суперкомп’ютерах реалізується багаторівнева модель паралельних обчислень двох основних рівнів паралелізму: верхнього – паралельно виконуються макрооперації (підзадачі – логічно незалежні частини алгоритмів) та нижнього – розпаралелення виконання кожної з макроопераций [78, 79].

Перший рівень моделі паралельних обчислень (верхній рівень, MIMD-модель) – паралелізм процесів (process level parallelism, PLP), за яким процеси паралельно виконують підзадачі, використовуючи як розподілену, так і загальну пам’ять CPU, синхронізуючи обчислення та обміни даними. Для розпаралелювання процесів на розподіленій пам’яті, найбільш ефективною є система MPI [80], а на загальній пам’яті – OpenMP [81].

Другий рівень (нижній рівень, SIMD-модель) – паралелізм потоків (thread level parallelism, TLP) – розпаралелювання макроопераций, використовуючи декілька потоків і спільну пам’ять.

Використання розподіленої пам’яті створює певні проблеми щодо обмінів даними між процесами – операцій, які залежать від реалізації (встановленого на комп’ютері) стандарту MPI та за тривалістю можуть значно перевершувати арифметичні операції, операції звернення до пам’яті. Отже, при створенні алгоритмів для розв’язування задач необхідно передбачати таке розміщення даних в пам’яті процесорів, при якому співвідношення пересилання 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).

Використання спільної пам’яті та середовища OpenMP усуває проблему обмінів даними між потоками, але створює проблеми з правильним використанням даних, вимагаючи додаткових синхронізацій та обмінів даними між основною пам’яттю та кешами різних рівнів. Крім того, середовище OpenMP краще підходить для синхронного виконання одних і тих же підзадач з різними даними через паралельні потоки, але реалізація асинхронного виконання різних макроопераций представляє значні труднощі. У такому середовищі ефективне виконання паралельних обчислень також вимагає рівномірного (збалансованого) завантаження використовуваних потоків. Якщо кількість інформаційних зв’язків (комунікаційних взаємодій) між підзадачами достатньо велика (порівняно з виконанням арифметичних операцій), або якщо ті самі підзадачі потрібно виконувати паралельно з різними даними, тоді рекомендується реалізувати відповідний алгоритм в середовищі OpenMP. Використання бібліотеки MPI

ефективне, коли кількість інформаційних зв’язків в алгоритмі мінімальна і виконуються переважно арифметичні дії.

Використання графічних процесорів дає змогу значно підвищити продуктивність обчислень. Графічний співпроцесор складається з тисячі однорідних ядер із загальною пам’яттю. Для розпаралелення обчислень використовуються технології NVIDIA CUDA [82] або OpenCL (Open Computing Language) від Apple [83], що здійснюють доступ до набору інструкцій відеокарти та керують її пам’яттю при організації паралельних обчислень. Але графічний процесор не має засобів прямої взаємодії з пристроями введення-виведення (крім монітора), а також доступу до оперативної пам’яті комп’ютера. Тому управління графічним процесором здійснюється засобами технології CUDA тільки через центральний процесор. Виникають проблеми пов’язані з узгодженням розподілу обчислювальних ресурсів між CPU та GPU, оптимізацією комунікаційних витрат, а також ефективного використання оперативної пам’яті різних рівнів.

На GPU добре розпаралелюються: задачі, які мають паралелізм за даними, тобто мають одну і ту ж послідовність однаково складних обчислень, що застосовуються до різних даних; задачі, які не потребують глобальної синхронізації; задачі, у яких кількість арифметичних операцій набагато більша у порівнянні з кількістю доступу до пам’яті. Обчислення розпаралелюється на велику кількість потоків. Отже, ті обчислення, на які CPU витрачає сотню тактів, графічний процесор здатний виконати за один такт.

Розглянемо більш детально елементи комп’ютерного середовища, які впливають на ефективність реалізації алгоритмів для розв’язування СЛАР.

Змінна топологія міжпроцесорних зв’язків. Ефективність розв’язування обчислювальних задач на паралельних комп’ютерах у великій мірі залежить від особливостей зв’язків між процесорами CPU 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, які за тривалістю можуть значно перевершувати арифметичні операції та операції звернення до пам’яті. Отже, необхідно передбачати такі комунікаційні зв’язки між обчислювальними елементами CPU, за якими співвідношення їх пересилань та арифметичних операцій, що одночасно виконуються, буде збалансованим і мінімізуватиме загальний час виконання програми. Розрізняють: одновимірні топології (лінійний масив); двовимірні топології (кільце, зірка, дерево, грати); тривимірні топології (повнозв’язна топологія, хордальне кільце); гіперкубічна топологія [10].

Комп’ютерне кешування. Процесори сучасних суперкомп’ютерів з багаторівневою комп’ютерною моделлю мають кеш-пам’ять – надшвидку статичну пам’ять. В 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 значний приріст продуктивності обчислень.

Найчастіше сучасні комп’ютери мають кеш-пам’ять двох або трьох рівнів. Кеш першого рівня (L1) – найбільш швидкий рівень кеш-пам’яті, який працює безпосередньо з ядром процесора і є буфером між процесором та кеш-пам’яттю

другого рівня. Кеш другого рівня (L2) – більш масштабний, ніж перший, але має менші «швидкісні характеристики». Відповідно, служить буфером між рівнем L1 та L3. Кеш третього рівня (L3) – повільніший, ніж два попередні, проте математичні операції виконуються набагато швидше, ніж на оперативній пам’яті. На відміну від L1 та L2, які поділяються на кожне ядро, 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’язування задач лінійної алгебри на паралельних комп’ютерах, в тому числі і гібридної архітектури, з метою ефективного використання обчислювальних ресурсів як з погляду ефективного використання кеш-пам’яті на CPU, так і з погляду швидкодії багатопоточного виконання однотипних арифметичних операцій на GPU є блочні алгоритми [10, 78, 79].

Ідентифікація та класифікація матриць засобами штучного інтелекту. Математичне моделювання процесів, які виникають в аналізі складних багатокомпонентних середовищ, показало, що вони досить часто зводяться до розв’язування СЛАР з розрідженими матрицями невизначеної структури та надвеликих порядків. В цьому випадку пропонується реалізація автоматичного розпізнання 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 виду («портрету») матриці [84, 85].

Структурна регуляризація, декомпозиція та зберігання розріджених матриць. Структура ненульових елементів (структура розрідженої матриці) визначається нумерацією невідомих задачі і може бути як регулярною (до прикладу, стрічковою, профільною, блочно-діагональною з обрамленням тощо), так і нерегулярною (довільною) [10, 40–44].

Існує ціла низка методів, які дозволяють регулювати заповнення матриці під час реалізації розв’язування задач на комп’ютерах. Розроблено як методи загального призначення, так і ті, що орієнтовані на матриці конкретного виду [40–46]. В залежності від поставленої задачі та структурних особливостей матриці, можна оптимізувати її портрет шляхом зміни ширини стрічки, зменшення профілю матриці, корегування загальної кількості заповнення, зведення структури до певного регулярного виду.

Декомпозиція (схеми розподілу) розріджених матриць. Багато методів розв’язування задач лінійної алгебри базуються на розвиненні матриці в добуток матриць стандартних виглядів, до прикладу, нижньої і верхньої трикутних матриць. Алгоритми таких розвинень характеризуються поступовим зменшенням від кроку до кроку розміру оброблюваної частини матриці. Тому важливо забезпечити приблизно однакові об’єми обчислень, обмінів та синхронізацій, що виконуються кожним процесом або потоком на паралельній моделі обчислень, тобто виключити вплив ефекту Гайдна [10]. Як показали дослідження, достатньо хорошу збалансованість завантаження процесів (потоків) забезпечують паралельні алгоритми, в яких використовуються так звані циклічні схеми розподілу і обробки матриць [10].

Схеми зберігання розріджених матриць. Існують різні схеми зберігання, які

відрізняються способом використання нулів. У деяких випадках допускається зберігання частини нулів в обмін на спрощення схеми зберігання; в інших − використовуються всі нулі матриці; в третіх − нулі взагалі не використовуються. Вибір формату зберігання, природно, впливає на запити до пам’яті, а отже істотно впливає на ефективність програмних реалізацій алгоритмів обробки розріджених матриць (до прикладу, розвинення матриць). Найчастіше використовуються такі формати зберігання елементів розріджених матриць: координатний формат, розріджений рядковий або стовпчиковий формат, формат ELLPaCK, гібридний формат.

Використання змінної розрядності. Як вже зазначалося, щоб забезпечити достатню точність для практичних задач з наближеними даними в ряді випадків необхідно використовувати підвищену комп’ютерну розрядність. Водночас, використання даних задачі у форматі з одинарною точністю дозволяє зберігати вдвічі більше даних на кожному рівні ієрархії пам’яті, включаючи кеш-пам’ять та регістрову пам’ять. В науковій роботі [86] відмічається, що оскільки багатоядерні комп’ютери (в тому числі потужні персональні комп’ютери) продовжують набирати обертів у світі високопродуктивних обчислень, доцільно переформулювати або розробити нові алгоритми для розв’язування задач лінійної алгебри при використанні змінної розрядності (одинарної та подвійної), щоб більш ефективно використовувати багатоядерність процесорів. В цих алгоритмах факторизація матриць СЛАР здійснюється послідовним її виконання над невеликими квадратними блоками матриці (плитками). Причому окремі обчислення над плитками виконуються на змінній розрядності (одинарній та подвійній) [70–72].

При моделюванні процесів в різних предметних областях виникають СЛАР з різними структурами матриць. Тому в ІСКМ використовуються адаптивані алгоритми для розв’язування цих задач з матрицями стрічковими, блочно-діагональними, хмарочосними. Ці алгоритми було апробовано на парктичних задачах [87–89, 91].

1.5 Інтелектулізація алгоритмічно-програмного забезпечення високопродуктивних обчислень з лінійної алгебри на сучасних суперкомп’ютерах

Велика різноманітність існуючого програмного забезпечення, реалізованого для суперкомп’ютерів на різних мовах програмування і функціонуючого під управлінням різних операційних та апаратних платформ, а також великі обсяги документації – все це для ефективного розв’язування прикладних задач потребує від користувачів значних інтелектуальних зусиль та часу. Крім того, як вже відмічалося, властивості комп’ютерної задачі можуть істотно відрізнятися від властивостей вихідної математичної задачі. Широко відомі бібліотеки та пакети програм розв’язування задач реалізують дослідницьку функцію частково, а врахування наближених даних покладається на користувача.

Такі проблеми можуть бути вирішені, якщо в комп’ютері дослідити властивості комп’ютерної моделі з наближеними даними та використати відповідний алгоритм отримання машинного розв’язку, що буде наближувати розв’язок математичної задачі з визначеною точністю при використанні необхідної розрядності комп’ютерних обчислень [1–4, 11, 12, 22, 51, 52].

Використання сучасних систем комп’ютерної математики. В наш час для вирішення питань ефективного математичного моделювання на сучасних комп’ютерах різної архітектури створено та використовуються системи комп’ютерної математики (СКМ). Це поняття включає сукупність як теоретичних та технологічних засобів, так і сучасних алгоритмічних, програмних і апаратних засобів, що надають можливість: формулювати задачу за допомогою спеціальної символьної мови в термінах математики; автоматизувати виконання як чисельних, так і аналітичних обчислень; розробляти складні обчислювальні алгоритми для розв’язування задач; застосовувати засоби візуалізації процесів та даних тощо [51, 52].

Найбільш відомими СКМ є програмні продукти, розроблені фірмами, такими як MathSoft, MathWorks, Waterloo Maple, Wolfram тощо. Найбільшу популярність мають програмні пакети Mathematica [29], MATLAB [30], Maple [31] за рахунок символьного процесора MathCAD [32]. Саме вони все частіше використовуються для розв’язання навчальних, інженерних, науково-дослідних задач у різних галузях природничих наук. Як відомо, чисельні алгоритми в цих програмних засобах створено на основі бібліотек програм: BLAS, LAPACK, MAGMA та інших. Для автоматичного розпаралелення задач на різних комп’ютерних архітектурах використовуються спеціальні програмні засоби. До прикладу, в Matlab використовується програмний засіб Parallel Computing Toolbox [53], завдяки якому обчислювальні задачі великих розмірів автоматично розпаралелюються на багатоядерних процесорах, графічних процесорах, комп’ютерних кластерах без програмування CUDA та MPI. Таким чином, користувачі звільняються від проблем розпаралелення задач. Серед

недоліків цих СКМ користувачі відмічають, що реалізація чисельних алгоритмів (до прикладу, розв'язування нелінійних або диференціальних рівнянь) в них базується на точно заданих даних. Тому є випадки отримання неправильних результатів розв'язування, не надається інформація яким методом розв'язувалася задача. Документація, яка надається користувачам досить велика (до прикладу, обсяг документації Matlab досягає майже 5 тис. сторінок, що робить її важко доступною для огляду). Специфічні редактори коду програм також ускладнюють роботу з СКМ [54].

В останні роки розробляються та використовуються інтелектуальні системи комп'ютерної математики (ІСКМ) які призначені для автоматизації дослідження та розв'язування задач на сучасних суперкомп'ютерах. Ці комп'ютерні системи створюються на основі сучасних засобів інтелектуалізації – баз знань (knowledge-based systems) про предметні області та методів штучного інтелекту (нейронні мережі, Data Mining тощо). Інноваційний напрям їх створення полягає в реалізації дослідження математичних моделей адаптивним способом на основі певних знань з предметної області та результатів комп'ютерного аналізу вихідних даних задачі, отримуючи в комп'ютері все більш уточнені знання про її властивості [57–66].

В Інституті кібернетики імені В.М. Глушкова на протязі десятків років проводяться дослідження щодо створення інтелектуальних комп'ютерів та інтелектуального прикладного програмного забезпечення. В цьому напрямку є досить вагові результати, які підвищили інтелектуальність комп'ютерних засобів в багатьох сферах та поліпшили якість математичного моделювання процесів з різних предметних областей [2, 11, 12, 14, 67–69].

В сучасних умовах швидкого зростання обсягів та складності задач в різних предметних областях, які підлягають математичному моделюванню, а також розмаїття архітектурних та технологічних особливостей існуючих суперкомп’ютерів проблеми інтелектуалізації алгоритмічно-програмного забезпечення для ефективного розв’язування СЛАР на різних моделях розпаралелення є досить актуальними. Основна мета інтелектуалізації – перекласти на комп’ютер якомога більше функцій щодо дослідження моделі, які раніше виконував користувач самостійно.

1.6 Постановка завдання дослідження

− розробити математичний апарат комп’ютерного дослідження математичних властивостей СЛАР з використанням багаторозрядної арифметики для забезпечення достовірності розв’язків на різних моделях паралельних обчислень;

− розробити і обґрунтувати адаптивний алгоритм для дослідження та розв’язування СЛАР з блочно-хмарочосними матрицями та наближеними даними на різних моделях паралельних обчислень гібридних комп’ютерів;

− розробити принципи, архітектуру та склад ІСКМ для автоматичного дослідження і розв’язання СЛАР в тому числі з матрицями розрідженої структури та наближеними даними на різних моделях паралельних обчислень;

− програмно реалізувати ІСКМ для дослідження та розв’язування СЛАР з матрицями різної структури та наближеними даними для гібридних комп’ютерів;

- провести експериментальне дослідження функціональних можливостей ІСКМ на комп’ютерах гібридної архітектури (до прикладу, суперкомп’ютерах родини кластерів СКІТ) на практичних задачах;
- інтегрувати ІСКМ у математичне забезпечення суперкомп’ютера родини кластерів СКІТ.

РОЗДІЛ 2

КОМП'ЮТЕРНЕ ДОСЛІДЖЕННЯ ТА РОЗВ'ЯЗУВАННЯ ЛІНІЙНИХ СИСТЕМ З НАБЛИЖЕНИМИ ДАНИМИ НА СУЧАСНИХ ПАРАЛЕЛЬНИХ КОМП'ЮТЕРАХ

Для розв'язування надскладних задач великих обсягів, які виникають в різних предметних областях, необхідно використовувати високопродуктивні комп'ютери, архітектури яких постійно ускладнюються. Ефективне використання суперкомп'ютерів потребує розробки нових підходів до побудови ефективних моделей паралелізму та їх відображення в існуючих або створених алгоритмах розв'язування задач [1–9]. Неповнота урахування функціональних можливостей паралельного комп'ютера в алгоритмах розв'язування задач може значно знизити ефективність його використання.

Крім того, в створюваних алгоритмах для розв'язування СЛАР необхідно враховуваkти наближений характер вихідних даних та проводити попереднє дослідження їх математичних властивостей [10, 11, 14, 70].

На всіх без виключення комп'ютерах на представлення будь-якого числа відводиться лише фіксована кількість розрядів. Тому після виконання кожної комп'ютерної операції результат «обрізається» до потрібної довжини. Тобто здійснюється заокруглення чисел введених в комп'ютер. Незважаючи на те, що помилки заокруглення окремих операцій можуть бути дуже малі, вони можуть змінити радикально математичні властивості задачі.

Важливим чинником, що пояснює вплив помилок заокруглення комп’ютерних операцій на остаточний результат розв’язування СЛАР, є особливості реалізації математичних операцій [10, 43]:

- континуум всіх дійсних чисел в комп’ютері апроксимується скінченною множиною скінченних дробів (вже при введенні числових даних в комп’ютер виникають похибки заокруглення, які визначаються так званим macheps);
- феномен «машинного нуля» породжує ряд труднощів при реалізації обчислювальних алгоритмів (будь-який сучасний комп’ютер має найменше додатнє число, яке може бути в ньому представлено, і всі числа, менші за абсолютною величиною від цього числа, замінюються нулем);
- арифметичні операції на комп’ютері відрізняються від математичних: закони асоціативності і дистрибутивності не виконуються на жодному сучасному комп’ютері, а закони комутативності в операціях з плаваючою комою виконуються тільки при правильній процедурі заокруглення.

Похибку у розв’язку СЛАР, яка обумовлена похибкою в вихідних даних, називають спадковою похибкою. Якщо спадкова похибка розв’язку велика, то отриманий математичний розв’язок може не мати фізичного змісту. Для зменшення спадкової похибки необхідно або підвищити точність задання вихідних даних або переформулювати задачу математичного моделювання щодо інших параметрів.

Обчислювальна похибка для прямих методів розв’язування СЛАР виникає внаслідок заокруглень в ході реалізації алгоритму розв’язування задачі в комп’ютері та наближеного характеру методу. Одним із способів мінімізації

помилок, пов'язаних із заокругленням в комп'ютері є подальше збільшення розрядності комп'ютерних обчислень [1, 70–72].

Для забезпечення достовірності отримуваних результатів розв'язування СЛАР виникає необхідність комп'ютерного дослідження їх математичних властивостей з урахуванням наближених даних та аналізу отримуваних результатів.

2.1 Комп'ютерне дослідження лінійних систем з наближеними даними

Розв'язування систем лінійних алгебраїчних рівнянь:

$$Ax = b\,, \tag{2.1.1}$$

де в загальному випадку A – прямокутна матриця розміру $m \times n$; b – матриця правої частини розміру $m \times q$, зводиться в класичному випадку до знаходження такого розв'язку x (матриці розміру $n \times q$), щоб рівняння (2.1.1) перетворювалося в тотожність. Якщо матриця A системи квадратна невироджена (тобто її визначник $|A| \equiv \det(A) \neq 0$), то розв'язок СЛАР (2.1.1) існує і єдиний.

Несумісні системи класичного розв'язку не мають, проте в цьому випадку можна знайти псевдорозв'язок x, який мінімізує евклідову норму $\|Ax - b\|$. Вектор x називається розв'язком за методом найменших квадратів або псевдорозв'язком СЛАР. В загальному випадку існує нескінченна множина псевдорозв'язків. Розв'язок, який має найменшу норму $\|x\|$, єдиний і називається нормальним псевдорозв'язком.

При розв’язуванні прикладних задач рідко виникають СЛАР з точними вихідними даними:

$$\overline{A}\,\overline{x} = \overline{b}\ . \qquad (2.1.2)$$

Найбільш типова є постановка задачі (2.1.1) разом з заданням відповідних похибок у вихідних даних:

$$\left\|\overline{A} - A\right\| = \left\|\Delta A\right\| \le \varepsilon_A \left\|\overline{A}\right\|, \qquad \left\|\overline{b} - b\right\| = \left\|\Delta b\right\| \le \varepsilon_b \left\|\overline{b}\right\|. \qquad (2.1.3)$$

При цьому передбачається, що структура матриці вихідної задачі (2.1.2) та збуреної задачі (2.1.1), (2.1.3) не змінюється, тобто, якщо вихідна матриця є симетрична, то і збурена залишається також симетричною, якщо вихідна – стрічкова, то і збурена – стрічкова, і т. д.

Тут ми розглядаємо випадок, коли A – квадратна матриця розміру $n \times n$; b – вектор правої частини СЛАР розміру n, x – розв’язок (вектор розміру n), ε_A, ε_b – максимальні відносні похибки елементів матриці та її правої частини відповідно.

Розв’язуючи СЛАР з наближено заданими вихідними даними, необхідно розглядати цілий клас систем рівнянь (2.1.1), (2.1.3), що має досить широку множину формально допустимих розв’язків. Тому, розв’язавши задачу (2.1.1), необхідно оцінити збурення розв’язку в залежності від збурення вихідних даних (2.1.3). Не завжди близкість елементів матриць A і $\overline{A}$ та правих частин b і $\overline{b}$ забезпечують достатню близкість розв’язків. Наприклад, при деякому збуренні в межах точності задання елементів матриці і / або правої частини несумісної точно заданої системи (2.1.2) отримана в комп’ютері збурена система

(2.1.1), (2.1.3) може виявитися сумісною і навпаки – сумісна СЛАР може перетворитися в несумісну [1].

Похибку розв'язку $\Delta x = \overline{x} - x$, яка обумовлена наближеним заданням вихідних даних, називають спадковою. Її числове значення залежить як від похибки вихідних даних (2.1.3), так і від властивостей матриці вихідної задачі (2.1.2).

Наближений розв'язок $\widetilde{x}$ системи (2.1.1), який отримується в результаті розв'язування задачі на комп'ютері, будемо називати комп'ютерним розв'язком задачі. Через похибки методу розв'язування та комп'ютерних обчислень отриманий розв'язок відрізняється від точного (математичного) розв'язку задачі (2.1.1). Різниця між цими розв'язками $\Delta\widetilde{x} = x - \widetilde{x}$ – це обчислювальна похибка.

Дослідження та розв'язування СЛАР з наближеними даними на комп'ютері складається з таких кроків [10, 70–72]:

- дослідження математичних властивостей комп'ютерної задачі (2.1.1), (2.1.3);
- аналіз достовірності розв'язку задачі – оцінка спадкової та обчислювальної похибок отриманого в комп'ютері розв'язку;
- розв'язування задачі алгоритмом, що відповідає відомим або виявленним математичним властивостям СЛАР з урахуванням архітектурних і технологічних особливостей комп'ютера.

В першу чергу комп'ютерного дослідження СЛАР з наближеними даними досліджується коректність постановки задачі (2.1.1), (2.1.3), тобто визначається існування, єдиність та стійкість класичного розв'язку.

Теоретичним критерієм коректності СЛАР (2.1.1), (2.1.3) є виконання умов:

$$\det(A) \neq 0, \qquad \|\Delta A\| \left\|A^{-1}\right\| < 1$$

для довільного збурення ΔA в межах (2.1.3), які гарантують існування, єдиність та стійкість класичного розв'язку задач в області зміни вихідних даних.

Комп'ютерне дослідження коректності зводиться до перевірки співвідношення:

$$1.0 + \gamma \neq 1.0, \tag{2.1.4}$$

де $\gamma = h^{-1}(A)$, $h(A)$ – число обумовленості матриці. Ця умова, яка виконується в арифметиці з плаваючою комою, означає що матриця невироджена в межах машинної точності (машинно невироджена), а умова $\left(\|\Delta A\| / \|A\|\right) h(A) < 1$, – що вона невироджена в межах точності задання вихідних даних.

При виконанні умов (2.1.4) розв'язок комп'ютерної задачі існує, єдиний і стійкий. Таку комп'ютерну задачу слід розглядати як коректно поставлену в межах машинної точності. В іншому випадку матриця системи може виявитися матрицею виродженою в межах машинної точності або погано обумовленою.

Для систем з прямокутними та виродженими матрицями обчислюється нормальний псевдорозв'язок, який, як вказувалося вище, у випадку сумісної системи співпадає з класичним розв'язком.

Відмітимо, що у формулі (2.1.4) для перевірки коректності комп'ютерної задачі входить величина γ, обернена до $h(A)$. Тому в разі виникнення великих чисел обумовленості в комп'ютері не настає переповнення за порядком.

Зникнення порядку для $h^{-1}(A)$ при великих числах обумовленості не фатально: комп'ютерний результат покладається рівним нулю, що дає можливість зробити правильний висновок про машинну виродженість матриці комп'ютерної задачі. У випадку, якщо в процесі розвинення (LU, LL^T) матриці СЛАР з'явилися нульові діагональні елементи, необхідно покласти γ рівним нулю, і тоді перша умова (2.1.4) дає можливість також зробити правильний висновок щодо виродженості матриці.

Таким чином, основним критерієм для визначення властивостей СЛАР, які впливають на достовірність розв'язку задачі, є число обумовленості $h(A)$ матриці системи.

Для невироджених матриць, число обумовленості обчислюється, використовуючи обернену матрицю A^{-1}

$$h(A) = \|A\| \left\|A^{-1}\right\|, \tag{2.1.5}$$

а в інших випадках – через псевдообернену матрицю $A^{\#}$

$$h(A) = \|A\| \left\|A^{\#}\right\|. \tag{2.1.6}$$

Для відносної спадкової похибки розв'язку задачі (2.1.1), (2.1.3) має місце оцінка [10]

$$\frac{\|x - \bar{x}\|}{\|x\|} \le \frac{h(A)(\varepsilon_A + \varepsilon_b)}{1 - h(A)\varepsilon_A} \tag{2.1.7}$$

або

$$\frac{\|x-\bar{x}\|}{\|\bar{x}\|} \le \frac{h(A)(\varepsilon_A+\varepsilon_b)}{1-\varepsilon_b} \qquad (2.1.8)$$

за умови $\|\Delta A\|\left\|A^{-1}\right\|<1$ та природному припущенні $\varepsilon_b<1$.

З оцінок (2.1.7) та (2.1.8) видно, що достовірність розв'язку системи визначається величиною числа обумовленості матриці та похибками вихідних даних задачі.

Оцінки мають мажорантний характер на всьому класі матриць. Проте, для невироджених матриць показана досяжність оцінки для першої матричної норми і тим самим доведена непокращуваність оцінки на класі невироджених матриць.

Для прямокутних ($m\times n$)-матриць повного рангу, тобто у випадку $r(A)=\min\{m,n\}$, де $r(A)$ – ранг матриці A, відносна спадкова похибка за умови $\|\Delta A\|\left\|A^{\#}\right\|<1$ оцінюється для $m > n$ так [73–77]

$$\frac{\|x-\bar{x}\|}{\|x\|} \le \frac{h(A)}{1-h(A)\varepsilon_A}\left(\left(1+h(A)\frac{\|r\|}{\|Ax\|}\right)\varepsilon_A+\varepsilon_b\right),$$

де $r=Ax-b$, а для $m < n$ так

$$\frac{\|x-\bar{x}\|}{\|x\|} \le \frac{h(A)(2\varepsilon_A+\varepsilon_b)}{1-h(A)\varepsilon_A}.$$

У випадку матриць неповного рангу задача знаходження нормального узагальненого розв'язку в загальному випадку некоректна. Якщо ранг матриці СЛАР (2.1.1) є $r(\bar{A})=k$, $A=U\Sigma V^T$ – сингулярне розвинення матриці з (2.1.1), (2.1.3), а $A^{(k)}=U\Sigma^{(k)}V^{(T)}$ (причому $\Sigma^{(k)}$ – квадратна матриця, перші

k діагональних елементів якої відмінні від нуля і співпадають з відповідними елементами матриці *A*, а всі останні елементи дорівнюють нулю), то нормальний узагальнений розв'язок $x^{(k)}$ системи $A^{(k)}x^{(k)}=b$ є ортогональною проекцією нормального псевдорозв'язку задачі (2.1.1), (2.1.3) на головний правий сингулярний підпростір матриці *A* розміру *k* [75, 76]. Тут необхідно розрізняти три випадки співвідношення рангів матриць $\overline{A}$ та *A*.

Якщо $\|\Delta A\|\left\|A^{\#}\right\|<1$ і $r(\overline{A})=r(A)$, то має місце оцінка

$$\frac{\|x-\overline{x}\|}{\|x\|}\le\frac{h(A)}{1-h(A)\varepsilon_A}\left(\left(2+h(A)\frac{\|r\|}{\|Ax\|}\right)\varepsilon_A+\varepsilon_b\right).$$

Якщо $\|\Delta A\|\left\|A^{\#}\right\|<0{,}5$ і $r(A)>r(\overline{A})=k$, то має місце оцінка

$$\frac{\left\|x^{(k)}-\overline{x}\right\|}{\|\overline{x}\|}\le\frac{h(A)}{1-2h(A)\varepsilon_A}\left(\left(2+h(A)\frac{\|r\|}{\|Ax\|}\right)\varepsilon_A+\varepsilon_b\right).$$

Якщо $r(\overline{A})>r(A)=l$, $\|\Delta A\|\sigma_l^{-1}<0{,}5$ і $\overline{x}^{(l)}$ – проекція нормального псевдорозв'язку задачі (2.1.2) на правий головний сингулярний підпростір матриці $\overline{A}$ розміру *l*, то має місце оцінка

$$\frac{\left\|\overline{x}^{(l)}-x\right\|}{\|\overline{x}\|}\le\frac{\sigma_{\max}\sigma_l^{-1}}{1-2\|A\|\sigma_l^{-1}}\left(\left(2+\frac{\sigma_{\max}}{\sigma_l}\frac{\left\|\overline{b}-\overline{b}^{(l)}\right\|}{\left\|\overline{b}^{(l)}\right\|}\right)\varepsilon_A+\varepsilon_b\right),$$

де $\bar{b}^{(l)}$ – проекція правої частини $\bar{b}$ на головний лівий сингулярний підпростір розміру l матриці $\bar{A}$, $\sigma_{\max} \equiv \sigma_1$ – максимальне, а σ_l – мінімальне відмінне від нуля сингулярне число матриці $\bar{A}^{(l)}$.

Викладене вище свідчить, що для аналізу властивостей комп'ютерної задачі з матрицею неповного рангу в умовах наближених вихідних даних фундаментальну роль відіграє визначення рангу матриці.

Рангом матриці в умовах наближених вихідних даних (ефективним рангом або δ-рангом) називають величину, що обчислюється за формулою

$$r_\delta(A) = \min_{\|A-B\| \le \delta} r(B). \qquad (2.1.9)$$

Це означає, що δ-ранг матриці дорівнює мінімальному рангу серед рангів всіх матриць B в околі $\|A-B\| \le \delta$. З [10] слідує, що якщо для $r(\delta)$ – δ-ранг матриці, то для її сингулярних чисел справедливі нерівності

$$\sigma_1 \ge \ldots \ge \sigma_{r(\delta)} > \delta \ge \sigma_{r(\delta)+1} \ge \ldots \ge \sigma_p, \qquad \text{де} \qquad p = \min\{m,n\}.$$

Практичний алгоритм для знаходження δ-рангу може бути реалізований так: знайти величину r, рівну найбільшому значенню i (i = 1, 2, ...), для якого виконується нерівність $\delta/\sigma_i < 1, \quad \sigma_i \ne 0$. Для аналізу значень рангу матриці в межах машинної точності величину δ можна покласти рівною $macheps \times \|B\|_\infty$ (*macheps* – найбільше число, для якого рівність $1.0 + macheps = 1.0$ є справедлива на комп'ютерах при обчисленнях з плаваючою комою [78]).

Сумарний вплив похибок заокруглень в прямих методах розв'язування СЛАР можна розглядати як відповідне еквівалентне збурення вихідних даних, тобто комп'ютерний (наближений) розв'язок $\tilde{x}$ системы (2.1.1) є точним для деякої збуреної системи, наприклад,

$$(A + dA)\tilde{x} = b + db \quad \text{або} \quad (A + F)\tilde{x} = b\,.$$

Тут *dA*, *db*, *F* – відповідні еквівалентні збурення, які залежать від методу розв'язування, порядка системи, довжини мантиси машинного слова та похибки переводу з десяткової системи в систему числення комп'ютера.

В роботах [10, 75, 79] представлено мажорантні оцінки обчислювальної похибки розв'язку СЛАР, які отримано прямим методом:

$$\frac{\|x - \tilde{x}\|}{\|x\|} \le \frac{h(A)(\delta_A + \delta_b)}{1 - h(A)\delta_A}, \tag{2.1.10}$$

$$\frac{\|x - \tilde{x}\|}{\|\tilde{x}\|} \le \frac{h(A)(\delta_A + \delta_b)}{1 - \delta_b} \quad \text{або} \quad \frac{\|x - \tilde{x}\|}{\|\tilde{x}\|} \le \|F\|\,\|A^{-1}\|, \tag{2.1.11}$$

де $\delta_A = \|dA\|\|A\|^{-1}$ та $\delta_b = \|db\|\|b\|^{-1}$

Оцінки (2.1.10), (2.1.11) свідчать про те, що вплив обчислювальної похибки особливо позначається при комп'ютерному розв'язуванні СЛАР з матрицями, які мають велике число обумовленості. Розгляд цих оцінок дозволяє зробити висновок про хорошу або погану обумовленість матриці СЛАР в зв'язку зі збуренням комп'ютерного розв'язку відносно сумарних похибок округлень. Погана «машинна обумовленість» тісно пов'язана з математичними

можливостями конкретного комп'ютера. Збільшуючи довжину машинного слова, завжди можна одержати комп'ютерний розв'язок, який буде досить близьким до математичного розв'язку задачі.

Оцінити обчислювальну похибку, тобто одержати інформацію про близькість комп'ютерного розв'язку $\tilde{x}$ до точного розв'язку x системи (2.1.1), а в ряді випадків зменшити похибку комп'ютерного розв'язку, можна шляхом його ітераційного уточнення [10].

Ітераційне уточнення розв'язку реалізується за наступною схемою:

$$x^{(0)} = x, \qquad r^{(s)} = b - Ax^{(s)}, \quad A \times \Delta x^{(s)} = r^{(s)},$$

$$x^{(s+1)} = x^{(s)} + \Delta x^{(s)}, \quad \text{де} \quad s = 0, 1, 2, \ldots .$$

При обчисленні $\Delta x^{(s)}$ використовується розвинення матриці, яке вже отримано при розв'язуванні СЛАР прямим методом, тому процедура ітераційного уточнення не потребує значного додаткового часу обчислень. Обчислення нев'язки $r_i^{(s)}$ виконується дещо з підвищеною довжиною машинного слова відносно основної розрядності. Наприклад, при виконанні основних обчислень з подвійною розрядністю обчислення нев'язки $r_i^{(s)}$ можна виконати з підвищеною розрядністю на регістрах комп'ютера.

Оцінка обчислювальної похибки розв'язку СЛАР визначається за формулою

$$E_{\text{обчис.}} <\approx \frac{\|\Delta x_1\|}{\|x_2\|},$$

де x_2 – наближення до точного розв’язку, яке отримано за один крок ітераційного уточнення.

Якщо для матриці системи виду (2.1.1), (2.1.3) не виконується умова (2.1.4), то така матриця вважається виродженою в межах машинної точності, тобто в околі машинної похибки може знайтись вироджена матриця. Але такий же ефект може давати і матриця з великим числом обумовленості відносно відповідної розрядності. Такі СЛАР доцільно дослідити та розв’язати з використанням підвищеної розрядності обчислень [39, 47, 70–72].

Невиконання умови (2.1.4) в комп’ютері на послідовності розрядних сіток свідчить про виродженість вихідної матриці. В цьому випадку можна знайти наближення до псевдорозв’язків СЛАР, використовуючи метод регуляризації на основі SVD-розвинення [73–77].

Виконання умови (2.1.4) на підвищенній розрядності у відповідності зі значенням оцінки числа обумовленості матриці вказує на те, що вихідна задача є погано обумовлена відносно попередньої розрядності і є можливість отримати наближення до єдиного розв’язку задачі (2.1.2), (2.1.3), використовуючи з підвищеною розрядністю, наприклад, алгоритми LU або LL^T – розвинення матриць.

Для реалізації комп’ютерного дослідження математичних властивостей лінійних систем з наближеними даними з матрицями різної структури (щільних, стрічкових, розріджених довільної структури) та розв’язування з оцінками достовірності отриманих розв’язків на багатоядерних комп’ютерах з графічними прискорювачами створено гібридні алгоритми на основі відомих ефективних

прямих та ітераційних методів, а саме: для щільних невироджених та вироджених матриць, прямокутних матриць довільного рангу, розріджених різної структури невироджених та вироджених матриць.

В кожному гібридному алгоритмі проводиться дослідження математичних властивостей СЛАР з наближеними даними та визначається їх відповідність вибраному гібридному алгоритму.

Як вже зазначалося, для обчислення числа обумовленості матриці за формулами (2.1.5) або (2.1.6) в комп'ютері необхідно обчислювати відповідно обернену матрицю A^{-1} або псевдообернену матрицю Мура–Пенроуза $A^{\#}$. Проте обчислення цих матриць потребує великих витрат часу та пам'яті комп'ютера, тому замість числа обумовленості доцільно використовувати його оцінку, яку в подальшому будемо позначати як cond$A$.

Дослідження математичних властивостей СЛАР з наближеними даними здійснюється на основі визначеної оцінки числа обумовленості матриці системи.

Для квадратної невиродженої матриці, в тому числі розрідженої структури, оцінка числа обумовленості обчислюється так [10]:

1) розв'язати СЛАР (тут, зазвичай, використовується трикутне розвинення матриці $A = LU$ або $A^T = U^T L^T$)

$$A^T y = e, \tag{2.1.13}$$

де e – вектор з компонентами ± 1, знак яких вибирається так, щоб норма вектора y була найбільшою;

2) розв'язати СЛАР

$$Az = y\,; \qquad (2.1.14)$$

3) обчислити оцінку числа обумовленості матриці

$$\mathrm{cond}A = \|A\|\,\|z\| / \|y\|; \qquad (2.1.15)$$

тут можна використовувати, наприклад, такі норми:

$$\|y\|_1 = \sum_{i=1}^{n} |y_i|, \quad \|z\|_1 = \sum_{i=1}^{n} |z_i|, \quad \|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^{n} |a_{ij}|.$$

Для прямокутної матриці довільного рангу доцільно використовувати спектральне число обумовленості:

$$\mathrm{cond}_s A = \frac{\sigma_{\max}}{\sigma_{\min}},$$

при обчислені якого використовується сингулярне розвинення матриці, де σ_{max} – найбільше сингулярне число, σ_{min} – найменше відмінне від нульового сингулярне число.

Розглянемо на деяких прикладах дослідження властивостей СЛАР, використовуючи підвищену розрядність.

Задача 1. Розв’язати систему рівнянь з такими вихідними даними:

$$A = \begin{pmatrix} 1002 & 1000 & -998 & 1000 \\ 1000 & 1002 & -1000 & 998 \\ -998 & -1000 & 1002 & -1000 \\ 1000 & 998 & -1000 & 1002 \end{pmatrix},$$

$b = (20042000 - 19962000)^T$, точний розв’язок системи $x = (1\ \ 1\ \ 1\ \ 1)^T$.

Фрагмент лістингу протоколу комп’ютерного дослідження та розв’язування задачі 1:

```
Method:
   Gaus decomposition

Double precision

Process of investigating   and   solving

!!! THE MATRIX IS MACHINE-SINGULAR !!!

COND ????
 PRECISION: 128

 SOLUTION
1.0000000000000000e+00    1.0000000000000000e+00
1.0000000000000000e+00    1.0000000000000000e+00
```

Отже матриця СЛАР, яка була введена в комп’ютер, виявилася виродженою в межах подвійної точності, а на розрядності 128 отримано наближення до єдиного розв’язку задачі з вихідною матрицею.

Для порівняння наводимо протокол розв’язування цієї задачі в режимі автоматичного вибору методу в MATLAB [15]:

```
warning: matrix singular to machine precision,
   rcond = 1.42109e-017
ans =
1.50000 0.50000 0.50000 0.50000
```

Тут, очевидно, отримано псевдорозв’язок машинно виродженої СЛАР (проекцію розв’язку вихідної задачі).

Задача 2. Розв’язати систему рівнянь з вихідними даними:

$$A=\begin{bmatrix} 8.0 & 1.0 & 6.0 & 7.0 \\ 1.0 & 4.0 & 2.0 & -3.0 \\ 6.0 & 2.0 & 11.0 & 4.0 \\ 7.0 & -3.0 & 4.0 & 10.0 \end{bmatrix}, \qquad b=\begin{bmatrix} 22.0 \\ 4.0 \\ 23.0 \\ 18.0 \end{bmatrix}.$$

Фрагмент лістингу протоколу комп’ютерного дослідження та розв’язування задачі 2:

```
    Method:
       Gaus decomposition

    Double precision

        THE MATRIX IS SINGULAR !!!

    Precision: 128

       THE MATRIX IS SINGULAR !!!
Method: singular value decomposition of a general matrix

    Double precision

    SOLUTION
    1.3333333333333344e+00    6.6666666666666763e-01
    1.0000000000000004e+00    6.6666666666666674e-01
Estimate of hereditary error of the solution: 3.10862e-15
```

Закінчення виконання задачі 2 алгоритмом LU – розвинення за методом Гауса на різній розрядності (double, precision 128) з повідомленням «THE MATRIX IS SINGULAR» свідчить про невиконання умов (2.1.4) і для double і для precision 128, тобто матриця вихідної СЛАР виявилася виродженою. В цьому випадку розв’язування було автоматично продовжено методом регуляризації на основі SVD-розвинення. На подвійній розрядності отримано наближення до нормального псевдорозв’язку задачі.

У Розділі 1 дисертації проведено аналіз різних існуючих способів реалізації обчислень з підвищеною комп’ютерною розрядністю, які дозволяють працювати з числами теоретично необмеженої довжини (розмір обмежений лише обсягом доступної оперативної пам’яті), але при цьому в декілька разів зменшується швидкість розв’язування задач.

В ІСКМ підвищена розрядність реалізується за допомогою бібліотеки програм GMP (GNU Multiple-Precision Library), що має великий набір оптимізованих процедур для підтримки обчислень з цілими, раціональними та дійсними числами з використанням довільної розрядності. Знаючи обумовленість матриці системи та точність обчислень на комп'ютері (macheps), можна визначити необхідну розрядність для отримання достовірного розв'язку [70–72].

В табл. 2.1 наведено значення *macheps* – найбільше число з плаваючою точкою, для якого виконується умова $1.0+macheps=1.0$ [10] і яке характеризує точність плаваючої арифметики для різної розрядності.

Таблиця 2.1 – Значення *macheps* для розрядності double, 64, 128 на процесорах Intel

Мова програмування	Довжина мантиси	*Macheps*
C++	double (53)	1.110e-16
	long double(64)	2.710505431213761085 02e-20
C++ з використанням GMP	128	1.469367938527859384960 92…e-39
	256	4.318084277547222312693 17…e-78

Приклад. Розглянемо СЛАР $Ax = b$, де A і b мають такий вигляд:

$$A = \begin{pmatrix} 0.1348531574\,394464 & 0.1878970588\,235294 & 0.1909117647\,058824 & 0.1779264705\,882353 \\ 0.1878970588\,235294 & 0.262 & 0.265 & 0.247 \\ 0.1909117647\,058824 & 0.265 & 0.281 & 0.266 \\ 0.1779264705\,882353 & 0.247 & 0.266 & 0.255 \end{pmatrix}$$

$$b = \begin{bmatrix} 0.3516 \\ 0.4887 \\ 0.5105 \\ 0.4818 \end{bmatrix}, \qquad x = \begin{bmatrix} 0.6662162162\,1616067387\,9806449043\,0572...e13 \\ 0.4016891891\,8907235069\,5216629824\,5477...e13 \\ 0.1665540540\,5399700518\,9401863978\,4241...e13 \\ 0.9772972973\,0279707257\,4940696301\,115...e12 \end{bmatrix}$$

Тут x – «точний» розв'язок системи (розв'язок порахований на розрядності 100).

В табл. 2.2 представлено результати розв'язування СЛАР, отримані за допомогою програми C++ з використанням функцій з бібліотеки GMP.

Результати експериментів свідчать про ефективність застосування функцій бібліотеки GMP для забезпечення достовірності результатів розв'язування СЛАР. Проте, як і при застосуванні інших існуючих способів реалізації комп'ютерних обчислень з підвищеної розрядності, швидкість розв'язування задач зменшується в десятки разів.

Таблиця 2.2 – Розв'язок СЛАР з розрядністю double та підвищеною розрядністю на MIMD-комп'ютері

Мова програмування	Довжина мантиси	Комп'ютерний розв'язок системи
C++	Double (53)	x_1=3.60239e+12 x_2=-2.17203e+12 x_3=-9.00599e+11 x_4=5.29764e+11
C++ з використанням GMP	64	x_1=**0.666**19910706694356510 9e13 x_2=**-0.40167**887337851497738e13 x_3=**-0.1665**49776766692724716e13 x_4=**0.9797**04569216725111902e12
	128	x_1=**0.666216216216160673879806**7836677102584254e13 x_2=**0.401689189189072350695216**8315835297097735e13 x_3=**0.16655405405399700518940 1**9476345873995266e13 x_4=**0.979729729730279707257494 5**61725193725136 2e12

2.2 Адаптивний паралельний алгоритм розв'язання лінійних систем з блочно-хмарочосними матрицями

В наш час дуже важливими є задачі аналізу міцності складних конструкцій та їх елементів у будівництві. Підвищення вимог до якості проектних рішень і

застосування нових конструктивних матеріалів вимагають розв’язання якісно нових задач.

Часто при розрахунку будівельних конструкцій виникають задачі з симетричними матрицями особливої структури. У верхньому трикутнику таких матриць ненульові значення розміщуються вертикальними смугами по кілька стовпчиків підряд (блоками), що на вигляд нагадує хмарочоси (рис. 2.1).

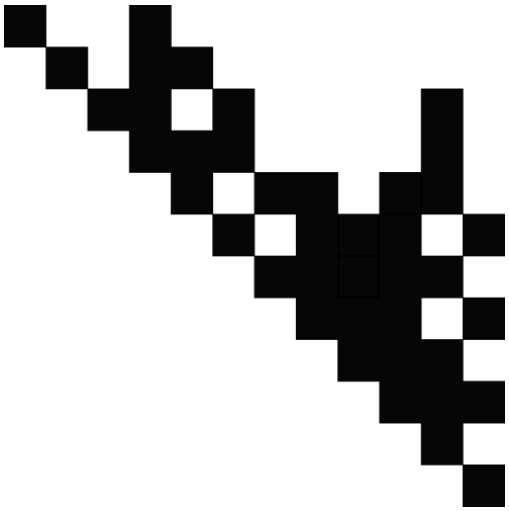

Рисунок 2.1 – Приклад блочно-хмарочосної структури

Таким чином, будь-який рядок верхнього трикутника складається з послідовних груп ненульових та нульових елементів. Оскільки ненульові елементи зберігаються групами, зникає необхідність зберігати індекси кожного елемента, що дає змогу зменшити розміри індексних масивів. У цьому полягає суть хмарочосної схеми.

Паралельний адаптивний алгоритми $U^{T}U$-розвинення ддя розв’язання лінійної системи. При розв’язанні СЛАР вирішальну роль у балансуванні завантаження процесів, враховуючи загальну кількість арифметичних операцій,

грає розподіл даних між процесами при виконанні трикутного розвинення матриці. Розподіл між процесами елементів трикутного розвинення розрідженої матриці визначає також розподіл решти даних, що задіяні у обчисленнях. Залежно від розподілу даних можна одержати ту чи іншу алгоритмічну реалізацію певного методу [89].

Для алгоритму $U^{\mathrm{T}}U$-розвинення симетричної матриці блочно-хмарочосної структури використано наступну схему обчислень**.** Розглянемо випадок блочно-розрідженої структури даних. Розглянемо комп'ютер з p CPU та спільною пам'яттю.

Нехай A – квадратна матриця порядку n. Розіб'ємо її на блоки розмірністю $s\times s$. Розміри сітки блоків $N\times N$, де $N=\left\lceil \frac{n}{s} \right\rceil$, при умові що n, націло ділиться на s, інакше $N=\left(\left\lceil \frac{n}{s} \right\rceil+1\right)$.

Для виконання факторизації матриці на k-ому кроці блочного розвинення використаємо наступне співвідношення

$$A^k=\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}=\begin{pmatrix} U_{11}^T & 0 \\ U_{12}^T & U_{22}^T \end{pmatrix}\begin{pmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{pmatrix}, \qquad (2.2.1)$$

де розміри блоків $A_{11}-s\times s$, $A_{12}-(N-ks)s$, $A_{22}-(N-ks)(N-ks)$, а блоки A_{12} та A_{22} враховують розріджену структуру підматриці A^k (блочної підматриці A).

Тепер можна застосувати метод $U^{\mathrm{T}}U$-розвинення до блоків на k-му кроці:

$$A_{11}=U_{11}^TU_{11}; \qquad (2.2.2)$$

$$U_{12} = \left(U_{11}^T\right)^{-1} \times A_{12}; \tag{2.2.3}$$

$$\tilde{A}_{22} = A_{22} - U_{12}^T U_{12}. \tag{2.2.4}$$

Зазначимо, що реалізація (2.2.1–2.2.4) на кожному кроці модифікує тільки ненульові блоки матриці *A*.

Для подальшого представлення алгоритму введемо додаткове позначення $NZ(k)$ – кількість ненульових блоків у *k*-му рядку блоків верхнього трикутника без діагонального блоку, A_{kj}, U_{kj} – ненульові блоки у *k*-му рядку блоків верхнього трикутника. Кожен з *p* потоків, що використовуються в обчисленнях, оброблятиме $\frac{NZ(k)}{p}$ стовпчиків блоків в матриці *A*.

Паралельний алгоритм U^TU-розвинення симетричної матриці блочно-хмарочосної структури реалізується наступним чином:

– Факторизуємо діагональний блок

$$A_{kk} = U_{kk}^T U_{kk} \tag{2.2.5}$$

– У всіх потоках модифікуємо позадіагональні блоки з A_{12}

$$U_{kj} = \left(U_{kk}^T\right)^{-1} A_{kj},\ j = k+1, \ldots, N \tag{2.2.6}$$

– У всіх потоках оновлюємо відповідні блоки з A_{22} за наступними співвідношеннями

$$\tilde{A}_{ij} = A_{ij} - U_{ki}^T U_{kj}, i = k+1, ..., N;\ j = i, ..., N; \tag{2.2.7}$$

Паралельні алгоритми розв'язання систем з блочно-трикутними матрицями. Один з етапів розв'язання систем з розрідженою блочно-хмарочосною матрицею є розв'язання систем з блочними трикутними матрицями. Розглянемо відповідні паралельні алгоритми, які дають змогу ефективно реалізувати обчислення, використовуючи декомпозицію даних, що була запропонована вище.

Паралельний алгоритм розв'язання системи з нижньою блочно-трикутною матрицею реалізується наступним чином:

Послідовно для кожного i-го $(i = 1,2, \dots, N-1)$ блочного рядку виконуємо наступні макрооперації

- розв'язуємо систему з нижньою трикутною матрицею

$$U_{ii}^T y_i = b_i$$

- паралельно у відповідних потоках модифікуємо k-ті частини вектора правих частин

$$b_k = b_k - U_{ik}^T y_i . k = \overline{i+1, N}$$

Для останнього рядка розв'язується система $U_{NN}^T y_N = b_N$.

Паралельний алгоритм розв'язування систем з блочною верхньою трикутною матрицею реалізується наступним чином:

- виконуємо розв'язання системи з верхньою трикутною матрицею

$$U_{NN} x_N = y_N .$$

Для i $i=\overline{1,N-1}$ у зворотному порядку виконуємо мікрооперації:

– у відповідних потоках модифікуємо y_i

$$y_i^* = y_i - U_{ik}x_k ; k=\overline{i+1,N};$$

– виконуємо розв’язання трикутних систем

$$U_{ii}x_i = y_i^* .$$

Оцінки ефективності та прискорення паралельних алгоритмів. Оскільки етап знаходження трикутного розвинення матриці найбільш складний по часу і кількості операцій сумарне прискорення програми буде визначатись саме прискоренням алгоритму розвинення. Для оцінки якості паралельних алгоритмів використовуються такі критерії, як коефіцієнти прискорення і ефективності, які відповідно можуть бути визначений наступним чином (див., напр., [10, 89])

$$S_p = T_1 / T_p, \qquad E_p = S_p / p, \tag{2.2.8}$$

де p – кількість процесів, що використовуються для розв’язування задачі, T_p – час розв’язування задачі на комп'ютері з використанням p процесів, T_1 – час розв’язування тієї ж задачі на гіпотетичному послідовному комп'ютері з швидкодією одного процесора і оперативною пам'яттю, яка дорівнює сумарній пам’яті, яка використовується p процесами.

Час виконання послідовного алгоритму для знаходження коефіцієнтів (2.2.1) можна обчислити так

$$T_1 = O_1\, t.$$

Час виконання паралельного алгоритму визначається так

$$T_p = \max\{\, T_{p,1}, T_{p,2}, \ldots, T_{p,p} \},\quad T_{p,i} = T_{\text{пар},i} + T_{\text{посл},i} + T_{\text{о},i} + T_{\text{с},i}\,.$$

Тут t – середній час виконання однієї арифметичної операції (додавання або множення) з плаваючою комою, O_1 – кількість таких арифметичних операцій у послідовному алгоритмі, $T_{пар,i}$ – час, що витрачається i-м процесом на паралельні арифметичні операції, $T_{посл,i}$ – час, що витрачається i-м процесом на послідовні арифметичні операції, $T_{о,i}$ – час, що витрачається i-м процесом на обмін даними з іншими процесорними пристроями, $T_{с,i}$ – час, що витрачається i-м процесом на синхронізації з іншими процесорними пристроями. Для багатопроцесорних (багатоядерних) комп’ютерів час T_p можна визначити і так

$$T_p = O_p\, t + O_o \tau_o + O_c\, \tau_c\,,$$

де O_p – максимальна кількість арифметичних операцій з плаваючою комою, що виконуються одним процесом у паралельному алгоритмі, t – час, необхідний для обміну одним машинним словом між двома процесами, O_o, τ_o – максимальний обсяг (кількість машинних слів) обмінів, що виконуються одним процесом, та відношення часу, необхідного для обміну одним машинним словом між двома процесами до середнього часу виконання арифметичної операції, τ_c – час, необхідний для синхронізації двох процесів, O_c, τ_c – максимальна кількість синхронізацій, що виконуються одним процесом, та відношення часу

необхідного для синхронізації двох процесів до середнього часу виконання арифметичної операції.

Теорема 1. Кількість операцій виконуваних послідовним алгоритмом оцінюється величиною $O_1 = \frac{Ns^3}{3} + 2s^3 * \sum_{k=1}^{N-1}\left(\frac{NZ(k)^2 + 3NZ(k)}{2} \right)$.

Доведення. Розглянемо алгоритм факторизації блочно-хмарочосної матриці. Враховуючи розбиття на блоки, ми отримаємо, що для кожного з N блочних рядків нам необхідно провести розвинення діагонального блоку (2.2.5), для кожного з $N-1$ блочних рядків $NZ(k)$ раз обчислюються макрооперації (2.2.6) для оновлення позадіагональних блоків верхнього трикутника розвинення та $\frac{NZ(k)^2 + NZ(k)}{2}$ раз виконати (2.2.7) для оновлення всіх блоків відповідної блочної підматриці.

Звідси випливають наступні оцінки кількості операцій, що виконуються алгоритмом в процесі обчислень:

- факторизація діагональних блоків – $\frac{Ns^3}{3}$ операцій;
- оновлення позадіагональних блоків – $2s^3 \sum_{k=1}^{N-1} NZ(k)$ операцій;
- оновлення значень у блоках підматриці – $2s^3 \sum_{k=1}^{N-1} \frac{NZ(k)^2 + NZ(k)}{2}$ операцій.

Просумувавши ці величини та спростивши вираз, ми отримуємо співвідношення $O_1 = \frac{Ns^3}{3} + 2s^3 * \sum_{k=1}^{N-1}\left(\frac{NZ(k)^2 + 3NZ(k)}{2} \right)$.

Теорема доведена.

Теорема 2. За умови рівномірної завантаженості процесорів обчисленнями кількість операцій необхідних для реалізації паралельного алгоритму визначається величиною

$$O_p = \frac{Ns^3}{3p} + 2s^3 * \sum_{k=1}^{N-1}\left(\frac{NZ(k)^2 + 3NZ(k)}{2p}\right).$$

Доведення. Використавши попередній розподіл даних та інформацію про кількість макрооперацій (2.2.5–2.2.7), знайдемо кількість операцій, що виконуються одним потоком при виконанні паралельного алгоритму.

Оскільки для виконання алгоритму в нас задіяно p потоків і архітектура комп'ютера у нас зі спільною пам'яттю, це дозволяє нам для факторизації діагонального блоку використати один з існуючих алгоритмів розвинення щільних матриць на одно вузлових комп'ютерах. Кількість операцій, що виконуються одним потоком буде оцінюватись величиною – $\frac{Ns^3}{3p}$ операцій.

Оновлення позадіагональних блоків проводиться незалежно у кожному з p потоків, відповідно виконується $2s^3\sum_{k=1}^{N-1}\frac{NZ(k)}{p}$ операцій

Оновлення значень у блоках підматриці також виконується у кожному з p потоків – $2s^3\sum_{k=1}^{N-1}\frac{NZ(k)^2 + NZ(k)}{2p}$ операцій

Просумувавши ці величини та спростивши вираз ми отримуємо співвідношення $O_p = \frac{Ns^3}{3p} + 2s^3 * \sum_{k=1}^{N-1}\left(\frac{NZ(k)^2 + 3NZ(k)}{2p}\right)$.

Теорема доведена.

Теорема 3. За умови рівномірної завантаженості процесорів обчисленнями Прискорення паралельного алгоритму оцінюється величиною

$$S_p \approx \frac{p}{1+2*\left(\frac{Ns}{p}+\sum_{k=1}^{N-1}\left(\frac{NZ(k)^2+3NZ(k)}{2p}\right)\right)\tau_c}.$$

Доведення. Для знаходження величини прискорення використаємо співвідношення (2.2.8). В ньому вже відомі O_1 та O_p. Оскільки алгоритм передбачає виконання в межах одного вузла і використовує спільну пам'ять, величиною O_o можна знехтувати, так як обміну даними фізично практично не відбувається. Кількість синхронізацій при виконані алгоритму можна оцінити величиною $2\left(\frac{Ns}{p}+\sum_{k=1}^{N-1}\left(\frac{NZ(k)^2+3NZ(k)}{2p}\right)\right)$, оскільки нам треба двічі робити синхронізації потоків, де $\frac{Ns}{p}$ – кількість синхронізацій при виконанні розвинення діагонального блоку, $\sum_{k=1}^{N-1}\left(\frac{NZ(k)^2+3NZ(k)}{2p}\right)$ – кількість синхронізацій, що виконуються при операції оновлення матриці розвинення.

Підставивши всі значення у формулу (2.2.8) та провівши відповідні спрощення отримаємо вираз

$$S_p \approx \frac{p}{1+2*\left(\frac{Ns}{p}+\sum_{k=1}^{N-1}\left(\frac{NZ(k)^2+3NZ(k)}{2p}\right)\right)\tau_c}$$

Теорема доведена.

Апробація паралельних алгоритмів для розв'язування СЛАР з хмарочосними матрицями на основі U^TU-розвинення. Розроблені паралельні алгоритми та відповідні програми було апробовано при розв'язанні задач розрахунку міцності будівельних конструкцій. Для тестів було обрано постановки задач: аналіз напружено-деформованого стану фундаменту будівлі; статичний розрахунок напружено-деформованого стану промислової будівлі; аналіз міцності конструкції багатоповерхового будинку.

Апробація проводилась на вузлах 8-вузлового кластеру СКІТ-5 з піковою продуктивністю 100 ТФлопс. 8-вузловий кластер СКІТ-5 з 192-ядерними вузлами на AMD EPYC 7642 з піковою продуктивністю 100 ТФлопс, інтегрований із загальним сховищем даних кластерного комплекса обсягом 200 ТБ; мережа передачі даних між вузлами Infiniband HDR 200 Гбіт/с; операційна система Ubuntu 20.04; кожен вузол має 256 ГБ оперативної пам'яті. Для організації паралелізму між потоками і присвоювання блоків хмарочосів відповідним потокам застосовувались: технологія OpenMP [81]; модулі бібліотеки Intel® MKL [90] – для ефективного виконання математичних операцій.

2.3 Висновки

У розділі 2 досліджено математичний апарат комп'ютерного дослідження та розв'язування СЛАР з наближеними даними при використанні багаторозрядної арифметики: дослідження коректності постановки, оцінок

спадкової та обчислювальної похибок, запропоновано реалізацію на основі багаторозрядної арифметики алгоритму автоматичної ідентифікації математичних властивостей задачі (виродженість, погана обумовленість). Вперше реалізовано в ІСКМ [70–72].

Розроблено паралельний алгоритм для розв’язання лінійних систем з блочно-хмарочосними матрицями. Проведено теоретичну оцінку прискорення паралельного алгоритму на спільній пам’яті та практичне дослідження на низці СЛАР з розрідженими матрицями хмарочосного формату, результати яких свідчать про високу ефективність запропонованого паралельного алгоритму [89].

РОЗДІЛ 3

АВТОМАТИЧНА ІДЕНТИФІКАЦІЯ ТА КЛАСИФІКАЦІЯ РОЗРІДЖЕНИХ МАТРИЦЬ НА ОСНОВІ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

Комп'ютерні моделі значної більшості практичних задач, отримані у результаті їх дискретизації (наприклад, методом скінченних елементів або скінченних різниць), зводяться до розв'язування СЛАР, матриці яких, як правило мають розріджену структуру. Матриця називається розрідженою, якщо кількість її елементів, відмінних від нуля, набагато менша від загальної їх кількості – n^2, де n – порядок матриці [44].

Для правильного вибору методу розв'язування цих задач та створення ефективних паралельних алгоритмів на їх основі необхідно враховувати такі математичні та структурні особливості розріджених матриць [46]:

- вид матриці – стрічкова, профільна, блочна, блочно-діагональна з обрамленням, хмарочосна, довільної структури тощо;
- порядок матриць, який може становити $O(10^5) - O(10^8)$;
- наближений характер елементів матриць і векторів, що виникають у процесі математичного моделювання: похибки дискретизації, похибки при введені в комп'ютер із заокругленнями тощо [10, 43, 44, 46].

Отже, для забезпечення достовірності комп'ютерних результатів розв'язування СЛАР з розрідженими матрицями та ефективного використання обчислювальних ресурсів необхідно:

– ідентифікувати в комп’ютері структуру розрідженої матриці та провести (у разі необхідності) регуляризацію – зведення довільної вихідної структури матриці до регулярної блочно-розрідженої (блочно-стрічкової, блочно-профільної, блочно-хмарочосної, блочно-діагональної з обрамленням тощо);

– вибрати найбільш ефективний (для даної архітектури комп’ютера, враховуючи регуляризовану структуру матриць) паралельний алгоритм розв’язування задачі;

– обрати відповідно до алгоритму схему декомпозиції (розподілу між процесорними пристроями паралельного комп’ютера) матриць та схему зберігання елементів матриць.

3.1 Методи структурної регуляризації, декомпозиції та зберігання розріджених матриць

Структура розрідженої матриці визначається нумерацією невідомих задачі і може бути як регулярною (наприклад, стрічковою, профільною, блочно-діагональною з обрамленням тощо), так і нерегулярною (довільною). На рис. 3.1 представлено структури матриць розрахункових задач, які виникають, наприклад, при моделюванні напружено-деформованого стану конструкцій.

Зведення розріджених матриць довільної структури до регулярного виду. Проблеми розв’язування задач лінійної алгебри з розрідженими матрицями та шляхи їх вирішення розглянуто в роботах [40-44]. В залежності від поставленої задачі та особливостей матриці, можна оптимізувати її портрет

шляхом зміни ширини стрічки, зменшення профілю матриці, корегування загальної кількості заповнення, зведення структури до певного виду.

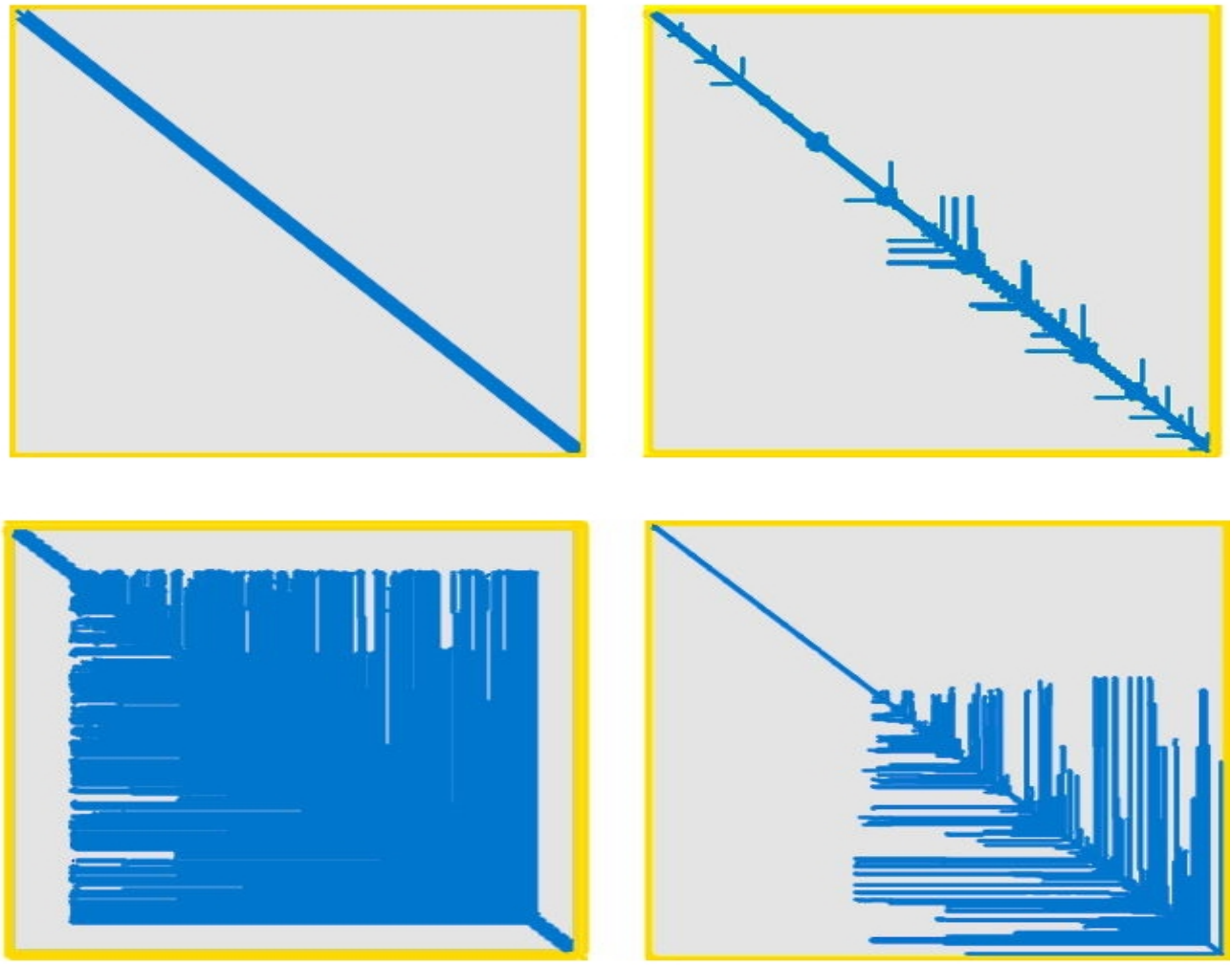

Рисунок 3.1 – Приклади структур розріджених матриць

Серед цих методів варто згадати такі:

- метод Катхілл-Маккі та метод фактор-дерев – використання цих методів забезпечує концентрацію ненульових елементів якнайближче до головної діагоналі, що дозволяє, як правило, представити матрицю у стрічковому або профільному вигляді;

- метод паралельних перерізів – використання цього методу дозволяє

представити матрицю як блочно-діагональну (з великими діагональними блоками) з обрамленням (порядок діагонального блоку обрамлення має бути значно меншим за порядки основних діагональних блоків) [40, 41];

- метод мінімальної степені – при використанні цього методу, як правило, отримують переупорядковану матрицю, для розвинення якої необхідно найменше арифметичних операцій; отриману структуру називають “хмарочосною” [92, 89] через те, що ця структура нагадує своїм виглядом хмарочоси Манхетена (рис. 3.2). Дискретні задачі з матрицями такого вигляду з’являються в задачах розрахунку будівельних конструкцій.

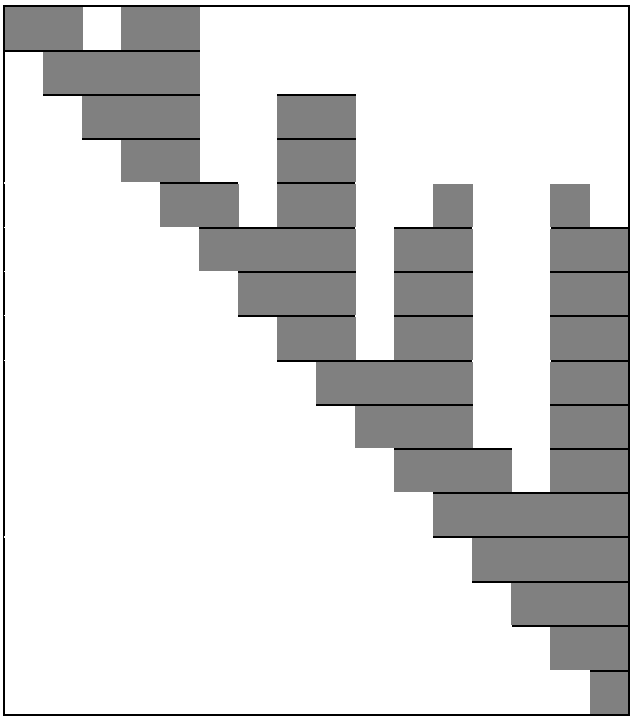

Рисунок 3.2 – Портрет верхнього трикутника «хмарочосної» матриці

Для ефективної реалізації в алгоритмах, наприклад матрично-матричних операцій розріджених матриць, доцільно спочатку так переупорядкувати (або одразу сформувати) їх, щоб переважна більшість ненульових елементів була

розташована у максимально щільних матричних блоках: тобто визначити нульові блоки (всі елементи тотожно дорівнюють 0) і ненульові (мають хоча б один ненульовий елемент). Бажано також, щоб ненульові блоки були якомога більше заповнені.

Якщо надалі передбачається використовувати розвинення розрідженої матриці у добуток трикутних матриць (напр., LU, LL^T тощо), то необхідно визначити блочно-розріджену структуру матриць розвинення, використавши символьне розвинення вихідної матриці. Після цього необхідно оптимізувати блочно-розріджену структуру вихідної матриці або матриць розвинення, використовуючи один з названих вище алгоритмів та замінивши елементи матриці в алгоритмах блоками.

В багатьох випадках доцільно об’єднати розташовані поряд (в рядку або стовпчику) ненульові позадіагональні блоки оптимізованої структури матриці в один прямокутний блок (плитку), який в подальшому обробляти як ціле. В деяких випадках доцільно оптимізувати структуру діагональних блоків з метою зменшення кількості арифметичних операцій.

Розглянемо ще один вид розрідженої матриці – блочно-діагональну матрицю з обрамленням (рис. 3.3). Дискретні задачі з матрицями такого вигляду з’являються, наприклад, при розв’язуванні крайових задач методом скінченних елементів або методом скінченних різниць, якщо область розбивається на підобласті та проводиться дискретизація, а невідомі дискретної задачі нумеруються в наступному порядку.

Спочатку невідомі, що належать тільки одній підобласті, послідовно по підобластях. Потім в такому ж порядку нумеруються невідомі, що належать двом

підобластям, далі трьом і так далі – ці невідомі формують обрамлення. Для ефективного розпаралелювання необхідно, щоб розміри діагональних блоків, які відповідають невідомим однієї підобласті, на порядок або більше перевищували розміри діагонального блоку обрамлення.

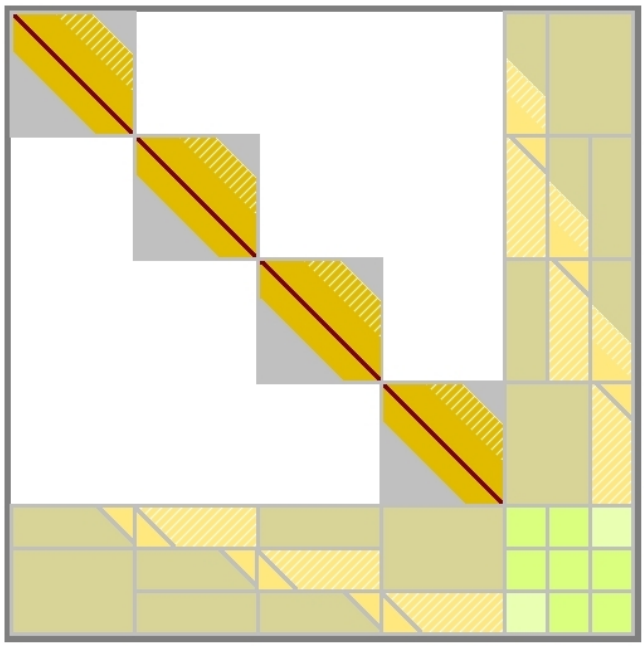

Рисунок 3.3 – Блочно-діагональна матриця з обрамленням (p = 4)

Декомпозиція (схеми розподілу) розріджених матриць. Багато методів та алгоритмів розв’язування СЛАР, що використовуються у моделюванні задач міцності конструкцій, базуються на розвиненні матриці задачі в добуток матриць стандартного вигляду, наприклад, нижньої і верхньої трикутних матриць. Як показали дослідження, паралельні версії цих алгоритмів з використанням так званих циклічних схем розподілу і обробки матриць (див. [10, 93, 94]) забезпечують достатньо хорошу збалансованість навантаження процесів (потоків).

Досить часто використовуються одновимірні блочно-циклічні схеми: циклічно між процесами або потоками верхнього рівня паралелізму розподіляються рядки або стовпчики ненульових блоків матриці, наприклад, якщо q-му процесу розподілено елементи рядків матриці з номерами $sr+1$, …, $sr+r$, то $(q+1)$-му процесу будуть розподілені рядки з номерами $s(r+1)+1$, …, $s(r+1)+s$, де s – кількість рядків у блоці (можна говорити про r-й і $(r+1)$-й рядки квадратних матричних блоків порядка s).

У разі регулярної структури розрідженої матриці паралельні алгоритми, які використовують одновимірну блочно-циклічну схему розподілу елементів матриці, можуть забезпечити приблизно однаковий об’єм обчислень і обмінів, що виконуються кожним паралельним процесом або потоком у будь-який час. Така регулярна структура є, перш за все, стрічковою структурою матриці за очевидної умови, що напівширина стрічки матриці $_m$ перевершує добуток $s \times p$, де p – кількість процесів або потоків. У разі регулярної профільної структури матриці системи, варіюючи значення s і p, можна також практично збалансувати завантаження паралельних процесів в кожний момент часу, якщо $c_e / n > sp$, де c_e – загальна кількість піддіагональних елементів у профілі матриці.

Для блочно-діагональних матриць з обрамленням циклічні схеми розподілу не дають можливості збалансувати завантаження процесів, а доцільно використовувати блочні схеми розподілу між обчислювальними елементами – кожний діагональний блок разом з відповідним позадіагональним блоком обрамлення розподіляється окремому процесу (або потоку). Також окремому

процесу може бути розподілено і діагональний блок обрамлення.

3.2 Використання штучних нейронних мереж для автоматичного визначення в комп'ютері структур та характеристик розріджених матриць

Штучні нейронні мережі – це складна система алгоритмів, яка використовується для вирішення задач штучного інтелекту, навчаючись на великій кількості даних та вдосконалюючи свою здатність вирішувати конкретні задачі. Завдяки нейронним мережам, комп'ютер отримує можливість аналізувати і навіть запам'ятовувати різну інформацію. Нейронні мережі також здатні не тільки аналізувати вхідну інформацію, а й відтворювати її зі своєї пам'яті. Іншими словами, штучна нейромережа це комп'ютерна інтерпретація мозку людини, в якому знаходяться мільйони нейронів та передають інформацію у вигляді електричних імпульсів [93, 94].

Реалізація автоматичного визначення структури матриць задачі за допомогою нейронної мережі та розв'язування необхідним гібридним алгоритмом складається з таких етапів:

- візуалізація матриці;
- визначення типу («портрету») матриці;
- визначення необхідного алгоритму розв'язування для типу матриці;
- дослідження та розв'язання задачі відповідним паралельним або гібридним алгоритмом.

Задача класифікації об'єктів полягає в призначенні кожному об'єкту одного з попередньо визначених класів або категорій на підставі його характеристик чи властивостей. Математично цю задачу можна сформулювати наступним чином:

Множина об'єктів (сукупність даних): Нехай X - множина об'єктів, кожен з яких описується вектором характеристик чи ознак. Об'єкти можуть включати зображення, текст, числові дані тощо.

Множина класів: Нехай C - множина можливих класів чи категорій, на які можуть бути класифіковані об'єкти. Кількість класів позначимо через $|C|$.

Функція класифікації: Нехай $f:X \to C$ - це функція класифікації, яка відображає об'єкт $x \subset X$ в один з класів $c \subset C$. Мета полягає в знаходженні оптимальної функції класифікації, яка найкраще визначає класи об'єктів на підставі їхніх ознак.

Навчання моделі: У багатьох випадках ця задача вирішується шляхом навчання моделі на основі тренувальних даних. Навчальний набір даних складається з пар (x_i, y_i), де x_i - ознаки об'єкта, y_i - його відомий клас.

Функціонал якості: Щоб оцінити якість моделі класифікації, вводять функціонал якості $Q(f)$, який вимірює відповідність між прогнозованими класами та реальними класами на тестовому наборі даних.

Оптимізація: Задача полягає в знаходженні функції класифікації f^*, яка максимізує функціонал якості: $f^* = \arg \max_f Q(f)$

Таким чином, математичною постановкою задачі класифікації є пошук оптимальної функції класифікації, яка краще всього визначає класи об'єктів на основі їхніх ознак.

Вхідні дані для задачі класифікації можуть бути представлені у вигляді графів, текстів, результатів запитів до бази даних тощо. За типами класів розглядають наступні класифікації.

Двокласова класифікація – найпростіший в технічному відношенні випадок, який служить основою для вирішення складніших завдань.

Багатокласова класифікація – коли кількість класів досягає багато тисяч (наприклад, при розпізнаванні ієрогліфів або злитого мовлення), то завдання класифікації стає істотно важчим.

Непересічні та пересічні класи – об’єкт може належати лише одному класу або одночасно декільком класам.

Нечіткі класи – потребують визначення ступеня належності об’єкта кожному з класів, зазвичай, це – дійсне число від 0 до 1.

Визначення типу («портрету») матриці є задачею машинного самонавчання, яка розв’язується за допомогою методів штучних нейронних мереж у вигляді навчання з учителем. В даному випадку дослідження проводяться з зображеннями і нечіткими класами.

Для визначення типу розрідженої матриці в алгоритмах розв’язування СЛАР побудовано згортковану нейронну мережу «Sparse Matrix Vision» [84, 85, 96], структуру якої представлено на рис. 3.4.

Основні її елементи такі: вхідний шар (Layer 1), згортковий шар (Layer 2), шар підвибірки (Layer 3), повнозв’язний шар (Fully Connected Layer); вихідний шар (Output Layer).

Основні етапи обробки вхідних даних в такій мережі:

Вхідний шар:

Вхідними даними є зображення у форматі JPEG розміром 224*224 пікселів, розділене на три канали: червоний, зелений і синій.

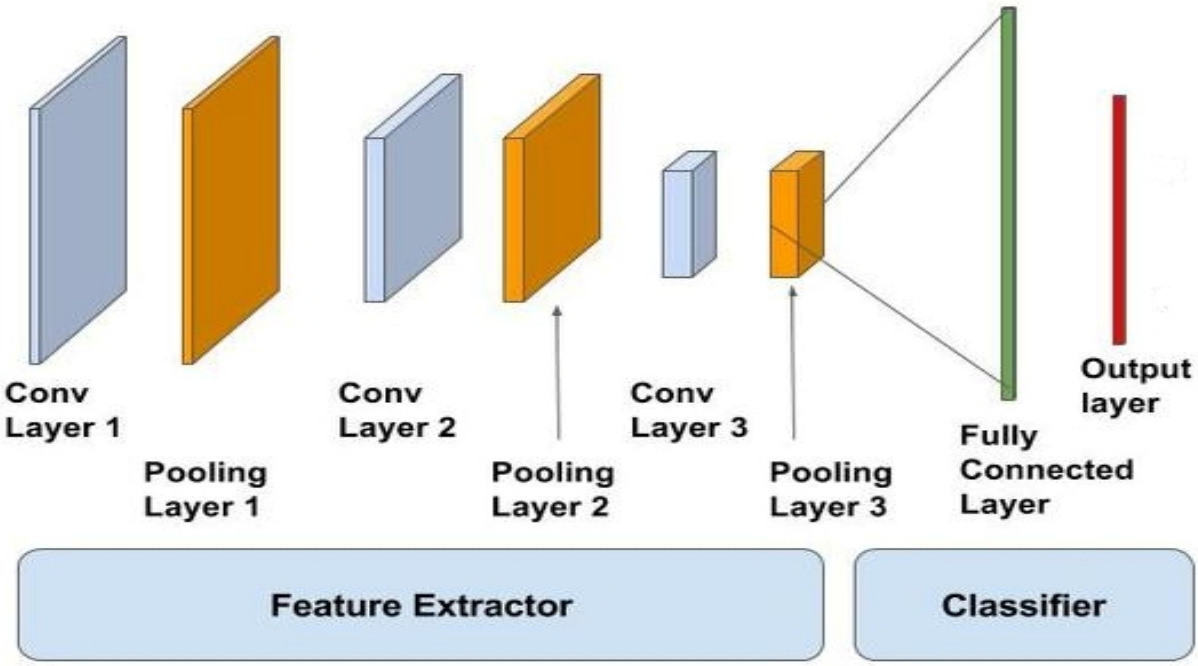

Рисунок 3.4 – Структура нейронної мережі «Sparse Matrix Vision»

Вхідний шар враховує двовимірну топологію зображення та складається з кількох карт, кожна з яких відповідає відповідному каналу зображення.

Нормалізація вхідних даних:

Значення кожного пікселя нормалізуються в діапазоні від 0 до 1.

Згортковий (Convolutional) шар:

Згортковий шар використовує набір ядер (фільтрів), які ковзають по попередній карті та виконують операцію згортки для визначення певних ознак об'єктів.

Ваги ядер задаються випадковим чином, і вони піддаються процесу оптимізації під час навчання.

Функція активації:

Після операції згортки застосовується функція активації ReLU (Rectified Linear Unit).

Шар підвибірки (Pooling):

Шар підвибірки зменшує розміри вихідних даних, ущільнюючи зображення та зберігаючи важливі ознаки.

Повторення згорткових і підвибіркових шарів:

Операції згортки та підвибірки можуть повторюватися для виявлення більш складних ознак та підготовки моделі для класифікації.

Пов'язані (Fully Connected) шари:

Вихідні дані оброблюються пов'язаними шарами, які пов'язують всі нейрони попереднього шару з кожним нейроном вихідного шару.

Функція активації може застосовуватися і в цих пов'язаних шарах.

Вихідний шар:

Вихідний шар пов'язаний з класами, які модель намагається класифікувати (у вашому випадку п'ять класів).

Функція активації для вихідного шару:

Використовується функція активації softmax для отримання ймовірностей для кожного класу.

Програмна реалізація та чисельний експеримент. Для навчання нейромережі та проведення чисельного експерименту було створено віртуальне оточення Python [97] із встановленими програмними пакетами Keras [98] та TensorFlow [99]. Для навчання використано набір із 2500 зображень структур матриць, в тому числі з колекції Флоридського університету (рис. 3.5).

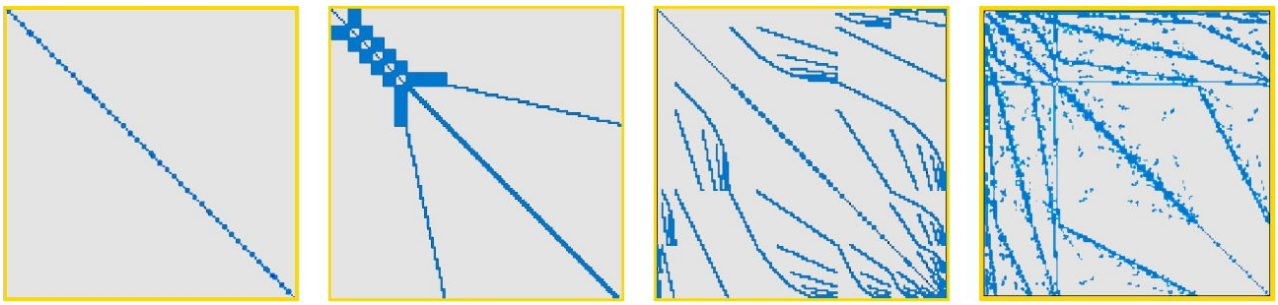

Рисунок 3.5 – Структури розріджених матриць з Флоридського університету

Експерименти проводились на обчислювальному вузлі супекомп'ютера СКІТ з наступними характеристиками: 2 центральні процесори Intel Xeon E5-2600 з частотою 2.6 ГГц; інтегрований із загальним сховищем даних кластерного комплексу обсягом 200 ТБ; мережа передачі даних між вузлами – Infiniband FDR 56 Гбіт/с; 128 ГБ оперативної пам'яті. У результаті навчання нейронної мережі отримано наступні графіки точності та втрат даних нейронної мережі (рис. 3.6) [84, 85].

Рисунок 3.6 – Точність та втрати даних нейронної мережі

Приклад виводу результатів навчання нейронної мережі:

```
0.8869001   0.00093775   0.00008755   .00000035   0.0000434
```

Числові дані вектора з 5 компонентів свідчать з ймовірністю 88 % приналежність матриці до профільного типу.

3.3 Висновки

У третьому розділі розроблено алгоритми автоматичного визначення структури розріджених матриць та їх регуляризації на основі методів машинного начання та нейромережевих технологій. Побудовано та навчено згорткову нейронну мережу «Sparse Matrix Vision». Виконана програмна реалізація та проведено чисельні експерименти, що показують ефективність застосування підходу до автоматичного визначення структурних характеристик матриць.

РОЗДІЛ 4

РОЗРОБЛЕННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ КОМП'ЮТЕРНОЇ МАТЕМАТИКИ (INPARSOLVER)

В сучасному світі активно розвивається напрямок створення прикладного алгоритмічно-програмного забезпечення на основі засобів штучного інтелекту (ШІ). У цьому напрямку досягнуто значні результати, які підвищують інтелектуальність комп'ютерних систем у різних сферах, таких як експертні системи, інтелектуальні пакети програм, інтелектуальні системи комп'ютерної математики, автоматизовані системи прийняття рішень та інше.

Як вказано в Розділі 1 дисертаційної роботи, серед ефективних алгоритмічно-програмних інструментів для математичного моделювання різноманітних процесів в науці та виробництві широке використання отримують інтелектуальні системи комп'ютерної математики. Вони розробляються на основі знанняорієнтованих принципів для відповідних предметних областей, використовуючи елементи штучного інтелекту.

4.1 Принципи, архітектура InparSolver та його складові частини

InparSolver – це ІСКМ, яка призначена для автоматичного дослідження та розв'язування задач математичного моделювання на багатоядерних комп'ютерах MIMD-архітектури з графічними процесорами. В дисертаційній роботі розглядається InparSolver для автоматичного дослідження та розв'язування СЛАР з матрицями довільної структури для гібридних

комп’ютерів [71, 72].

Принципи створення та використання InparSolver:

- постановка задачі з наближеними даними на мові предметної області;
- автоматизації процесу дослідження та розв’язування СЛАР з розрідженою структурою даних, що базується на формальній моделі предметної області і методах машинного навчання з використанням нейронних мереж [71, 72], [84, 85];
- оцінка достовірності комп’ютерного розв’язку, в тому числі з використанням змінної комп’ютерної розрядності при ефективному використанні обчислювальних ресурсів [87, 88];
- реалізація принципу прихованого паралелізму, який забезпечує роботу користувача на гібридному комп’ютері, як на комп’ютері стандартної однопроцесорної архітектури [10];

Прихований паралелізм передбачає:

- автоматичне визначення топології паралельного комп’ютера у відповідності до математичних властивостей задачі;
- автоматичне розпаралелення обчислень та даних, забезпечуючи рівномірне завантаження обчислювальних елементів, асинхронне виконання однотипних обчислень великих обсягів на GPU та копіювання необхідних даних між процесорами CPU та GPU.

Розроблення InparSolver за запропонованими принципами базується на основі бази знань про дану предметну область: формальної моделі предметної області та процедурних знань (методів автоматичної ідентифікації та класифікації матриць в комп’ютері засобами штучного інтелекту; машинних алгоритмів автоматичного комп’ютерного дослідження та розв’язування СЛАР

з наближеними даними при використанні багаторозрядної арифметики та довільної розрядності, нових адаптивних методах та паралельних алгоритмах розв'язування СЛАР з матрицями різної структури в змінному комп'ютерному середовищі на різних моделях розпаралелення гібридних комп'ютерів). На рис. 4.1. представлено архітектуру InparSolver.

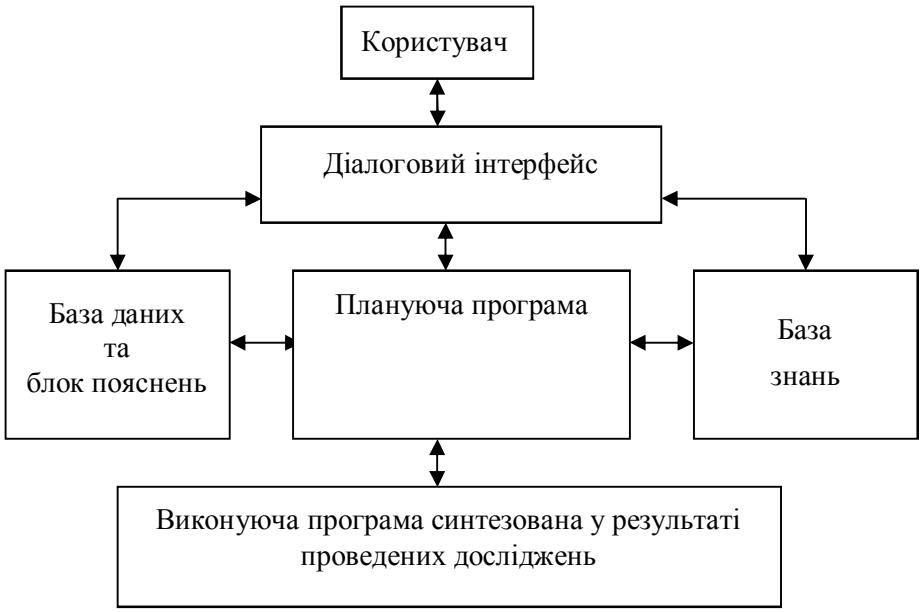

Рисунок 4.1 – Архітектура InparSolver

Складові частини InparSolver:

1) Діалоговий інтерфейс для спілкування з користувачами на мові предметної області;

2) База даних, до якої входять введені в комп'ютер дані СЛАР, результати дослідження задачі необхідними алгоритмами в числовій формі, отримані розв'язки задач. Ці дані використовуються в базі знань.

3) База знань з предметної області включає: формальну модель предметної області (декларативні знання) та процедурні знання (методи автоматичної ідентифікації та класифікації матриць в комп'ютері засобами штучного інтелекту; машинні алгоритми автоматичного комп'ютерного дослідження та

розв’язування СЛАР з наближеними даними при використанні багаторозрядної арифметики та довільної розрядності, адаптивні методи та паралельні алгоритми розв’язування СЛАР з матрицями різної структури для різних моделей паралельних обчислень.

4) Блок пояснень, в якому знаходиться інформація щодо функціональних можливостей InparSolver, глосарій використаних математичних термінів, приклади задання масивів даних тощо; накопичується інформація про хід обчислювального процесу задачі та використані алгоритми; результати аналізу достовірності отриманих розв’язків. Ця інформація надається користувачеві за необхідності.

5) Плануюча програма: програмні засоби, що реалізують автоматизацію всього процесу дослідження та розв’язування задачі на формальній моделі предметної області (графовій моделі), а також засоби штучного інтелекту (машинного навчання на основі штучних нейронних мереж) для автоматичного розпізнання структури матриць.

Комп’ютерна реалізація InparSolver включає:

− створення математичного апарату автоматичного дослідження СЛАР з наближеними даними;

−розробленння алгоритмічно-програмного забезпечення для дослідження та розв’язування СЛАР з матрицями різної структури [70–72, 87];

−розроблення та навчання згорткової нейронної мережу для автоматичного розпізнання в ІСКМ різних структур розріджених матриць, обробка яких присутня в значній більшості адаптивних алгоритмів обчислювальної

математики, що використовуються при математичному моделюванні фізико-технічних процесів [70–72, 87];

− побудова формальної моделі предметної області для класу задач з лінійної алгебри, використовуючи різні способи подання та застосування знань про конкретні задачі та алгоритми їх дослідження і розв’язування;

− побудова формальної моделі користувача та спілкування з ним, виходячи з аналізу моделі предметної області та форм спілкування для забезпечення ефективного розв’язування задачі;

− комп’ютерна реалізація експериментального зразка ІСКМ на суперкомп’ютері гібридної архітектури СКІТ-4.

4.2 Формальна модель предметної області

В InparSolver модель предметної області – це задачі, які пов’язані з розв’язуванням СЛАР з наближеними даними та взаємозв’язки між ними. Як відомо, на практиці СЛАР виникають при моделюванні об’єктів, що описуються диференціальними, інтегральними рівняннями або системами нелінійних алгебраїчних (трансцендентних) рівнянь. Можуть з’являтися як етап в задачах мінімізації або апроксимації функцій, а також як самостійний клас математичних задач. Використання методів скінченних різниць для розв’язування лінійних крайових задач з симетричними операторами приводить до СЛАР з симетричними додатно означеними або напівозначеними матрицями [100, 101]. Якщо вихідні диференціальні оператори несиметричні, то при дискретизації з’являються задачі з несиметричними розрідженими матрицями.

При дискретизації нестаціонарних рівнянь потрібно розв’язувати СЛАР для розріджених або профільних матриць (діагональних, трьохдіагональних, пятидіагональних, блочно-діагональних тощо).

При дискретизації методом скінченних елементів виникають задачі СЛАР з симетричними та несиметричними матрицями (щільними, стрічковими, профільними) [102, 103].

При рівномірній поліноміальній апроксимації виникають СЛАР з квадратними матрицями з повним заповненням. При розв’язуванні задач рівномірної апроксимації функцій багатьох змінних можуть виникати несумісні СЛАР з прямокутними матрицями [10].

База даних в InparSolver є структурованою. В ній зберігаються дані різних видів та призначення:

1) Дані для функціонування та використання штучної нейронної мережі розпізнання структур матриць. Ці дані зберігаються постійно та можуть доповнюватися при включенні до складу InparSolver нових алгоритмів з іншими структурами матриць.

2) Вихідні дані задач та точність їх задання, які можна зберігати, вилучати, виправляти та використовувати багатократно; дані про задачу, які використовуються лише в процесі розв’язування задачі, в цьому випадку термін збереження даних має атрибут часу; результати розв’язування задач, які можна візуалізувати та зберігати для подальшого використання.

Обсяги масивів даних СЛАР можуть бути досить великими, тому в InparSolver передбачено ефективні способи їх обробки.

До бази знань належать: інтелектуальні алгоритмічно-програмні засоби, що реалізують автоматизацію всього процесу дослідження та розв’язування задачі в змінному комп’ютерному середовищі на ефективній моделі розпаралелення; інтелектуальні алгоритмічно-програмні засоби для автоматичної ідентифікації та регуляризації структури вихідної матриці, використовуючи засоби машинного навчання – штучні нейронні мережі.

Для формального представлення в комп’ютері знань про предметну область доцільно застосовувати модульний аналіз. Під модульним аналізом розуміємо сукупність прийомів та методів розділення проблеми в цілому на окремі фрагменти (підзадачі – логічно завершені частини алгоритмів), встановлення між ними логічних зв’язків. З цією метою, на основі розробленого математичного апарата автоматичного комп’ютерного дослідження СЛАР з наближеними даними та розв’язування відповідними адаптивними алгоритмами, було встановлено взаємозв’язки між окремими програмними модулями, що реалізують визначені підзадачі, та послідовність їх виконання.

Зв’язки між модулями здійснюються на знаннях. Кожний модуль містить в собі різного виду знання: про можливість виконання обчислень, про вхідні дані та способи їх задання, про необхідні комп’ютерні ресурси, про результати досліджень тощо. Крім того, існують взаємозв’язки (семантика) між алгоритмами та програмними модулями щодо реалізації дослідження математичних властивостей задач та синтезу найбільш ефективного паралельного алгоритму і відповідної програми на необхідній конфігурації комп’ютера з оптимальної кількості CPU та GPU. Ці знання отримуються на основі теоретичних та практичних досліджень розроблених адаптивних

алгоритмів та програм для розв’язування СЛАР на різних моделях паралельних обчислень гібридних комп’ютерів.

Після того, як визначено вміст предметної області, проведена модуляція та встановлені всі логічні зв’язки на основі баз даних та знань можна побудувати її формальну модель. Для цього необхідно формалізувати знання про предметну область для представлення їх в комп’ютері.

Виділимо ті знання про предметну область, які підлягають формалізації:

- опис об’єктів предметної області (наприклад, параметри задачі, способи задання масивів даних – векторів та матриць, структура даних, виявлені в комп’ютері математичні властивості задач та їх застосування);
- відомості про програмні модулі – функціональне призначення, вхідні та вихідні дані, умови застосування, експлуатаційні характеристики тощо;
- база знань щодо способів визначення структури матриць та способів їх регуляризації, організації автоматичного дослідження та розв’язування задач з наближеними даними (можливі послідовності з кількох програмних модулів, в яких одні модулі замінюються / доповнюються іншими), знання щодо використання модулів для аналізу отриманих результатів;
- виводи, отримані в процесі розв’язування задачі (характеристика математичних властивостей СЛАР, відомості про достовірність комп’ютерних результатів, неповнота даних або неточність вхідних даних для розв’язування тощо).

Ці знання про формальну модель предметної області в InparSolver описано у вигляді декларативних та процедурних знань. Модульний аналіз надає можливість описати всю необхідну інформацію про предметну область як

сукупність трьох параметрів (a, r, b), де a та b – два об'єкти (програмні модулі), а r – відношення між ними, та застосувати для опису формальної графової моделі [58, 61, 62].

Таким чином, програмні модулі InparSolver та встановлені взаємозв'язки між ними дають можливість представити в комп'ютері реалізацію автоматичного дослідження та розв'язування СЛАР з наближеними даними у вигляді формальної моделі предметної області, до якої входять: виявлені в комп'ютері математичні властивості задач, ефективні паралельні алгоритми їх розв'язування в змінному комп'ютерному середовищі, способи задання вхідних даних, необхідна топологія комп'ютера тощо. А синтезована в процесі автоматичного дослідження задачі комп'ютерна програма є ланцюжок програмних модулів, склад та порядок виконання яких залежить від задіяних паралельних алгоритмів та результатів досліджень на кожному кроці математичного моделювання.

Як відомо, існують різні моделі формалізації знань: семантичні мережі, фрейми, нейронні мережі тощо [58, 61, 62].

Семантична мережа описується знаннями двох типів – декларативних та процедурних у вигляді графів. Декларативні знання, які характеризують аспект «ЗНАТИ ЩО», описують властивості об'єктів, визначення, факти з предметної області. Такі знання не мають тісних зв'язків з процедурами їх обробки. Процедурні знання – це знання, що мають відношення до методу логічного виводу – «ЗНАТИ ЯК». В якості базових понять тут виступають алгоритми та методи дослідження задачі. За допомогою процедурних знань визначається як їх можна використовувати для отримання нових знань про об'єкт чи процес з

предметної області. В семантичній мережі декларативні та процедурні знання представляються у вигляді орієнтованого графу, вершинами якого є декларативні та процедурні знання. Це можуть бути – визначення, властивості об’єктів, узагальнені поняття, формули, програми, функціональні дії тощо. Вершини графу з’єднуються між собою дугами, якщо між ними є зв’язки. Крім семантичних мереж досить часто дані та знання про об’єкти предметних областей формально описуються у вигляді фреймів. Фрейми представляють собою структуру даних, що дає цілісне уявлення про об’єкти, явища та їх типи у вигляді абстрактних образів. Структура фрейму записується у вигляді списку властивостей (слотів) об’єктів та різноманітних їх атрибутів [58, 61, 62].

В InparSolver для формалізації знань про предметну область використовувалися семантичні мережі , а для визначення структури матриць – нейронні мережі.

Для визначення взаємозв’язків між функціональними модулями застосовувалися узагальнено-логічні графи. Вершини–модулі в такому графі об’єднуються дугами двох видів – інформаційними та логічними. Інформаційні дуги визначають порядок слідування модулів при повній специфікації. Логічними дугами визначається порядок слідування на основі досліджень у вершинах-функціональних модулях або після деяких дій користувача щодо уточнення даних та умов задачі (ними навантажені дуги), тоді процес обчислень може продовжуватися в різних напрямках.

4.3 Формальна модель користувача

При створенні InparSolver передбачено побудову формальних моделей користувача та форм спілкування з ним. Було проведено дослідження можливих видів спілкування з користувачем на різних етапах роботи з InparSolver. Коротко розглянемо їх.

Вибір мети використання InparSolver. Користувач може ознайомитись з інформацією про призначення ІСКМ та його функціональними можливостями. Також за бажанням може отримати більш детальну інформацію про методи та паралельні алгоритми, що використовуються; опис методології дослідження задач з наближеними даними та аналізу комп'ютерних результатів; глосарій термінів з лінійної алгебри.

Вибір режиму розв'язування. Автоматичний режим розв'язування не передбачає спілкування з користувачем. В процесі дослідження та розв'язування конкретної задачі автоматично використовуються необхідні паралельні алгоритми різних методів, ефективна модель розпаралелення з оптимальної кількості процесорів та різної комп'ютерної розрядності для забезпечення достовірності комп'ютерних результатів діалоговий, за яким користувач самостійно. Інтерактивний режим дає можливість користувачу приймати участь у процесі розв'язування: вибирати кількість обчислювальних ресурсів, направляти хід обчислювального процесу за конкретно вказаними методами або призупити обчислення.

Введення вхідних даних задачі. Користувач може задавати вхідні дані з різних носіїв інформації – з екрану, заповнюючи спеціальні таблиці для масивів

даних, програмно або з файлу. InparSolver в динаміці аналізує введені дані, перевіряючи надані обчислювальні ресурси, правильність вводу даних тощо. При цьому користувач може уточнити параметри задачі, виправити помилки, ознайомитись з більш детальною інформацією щодо способів введення даних, проглянути конкретні приклади.

Виведення та пояснення результатів розв'язування. Результати розв'язування за бажанням користувача можуть бути виведені на різні носії інформації – на екран, на принтер, у файл. Крім розв'язку задачі, користувач отримує протокол процесу дослідження та розв'язування.

Надання допомоги користувачу на кожному етапі роботи з InparSolver. Передбачається надання необхідної інформації лише в разі необхідності. Причому поради InparSolver не повинні обмежувати побажання самостійних дій користувача (при використанні інтерактивного режиму), але мають позбавляти його від помилок або непродуктивного витрачання часу. Діалог з користувачем має не тільки забезпечувати ефективне розв'язування задачі, але також сприяти поповненню його знань з лінійної алгебри, паралельних обчислень на сучасних суперкомп'ютерах тощо [57–59, 104].

На рис. 4.2 узагальнено-логічним графом представлено в InparSolver фрагмент опису формальної моделі автоматичного дослідження та розв'язування СЛАР зі щільними матрицями.

Вершини цього графу наступних видів.

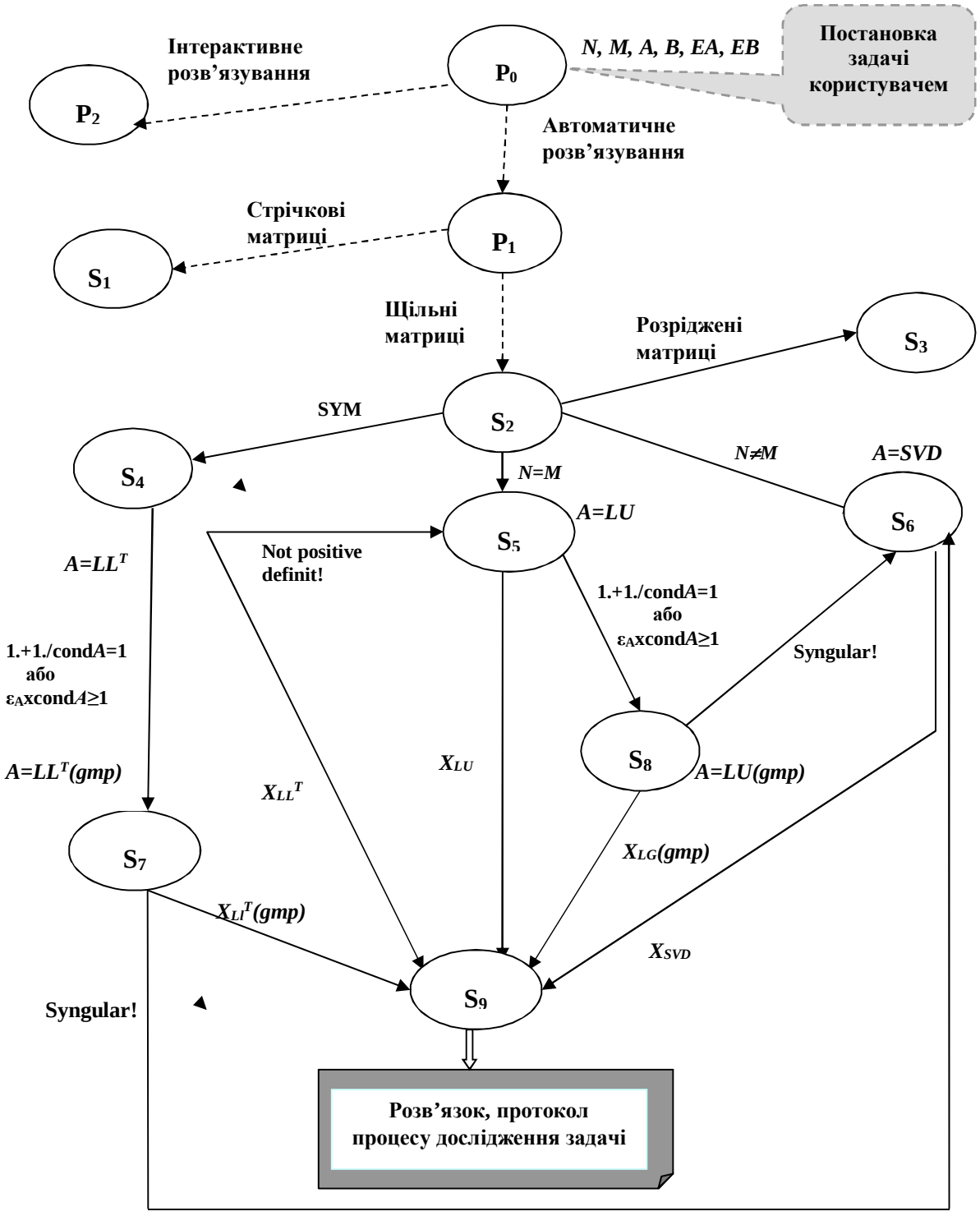

Рисунок 4.2 – Фрагмент формальної моделі автоматичного дослідження та розв'язування СЛАР зі щільними матрицями в InparSolver

Вершина (P_0) – введення вихідних даних СЛАР, (P_1) – вибір способу розв'язування задачі (автоматичний) та задання користувачем вихідних даних

(кількість рядків та стовпчиків матриці, масиви, що містять елементи матриці та правих частин, а також точність задання елементів СЛАР).

Вершина-перемикач (P_2) на графі інтерактивного розв'язування СЛАР.

Вершини логічного висновку (S_1), (S_2), (S_3) – в InparSolver автоматично отримується інформація щодо структури матриці СЛАР, яка визначається на основі результатів використання методів та програмних засобів штучного інтелекту – нейронних мереж.

Вершини-функціональні модулі (S_4, S_5, S_6), що використовуються для початкового дослідження введеної в комп'ютер СЛАР, синтезованою програмою на ефективній моделі розпаралелення гібридного комп'ютера. Якщо в результаті такого дослідження виявляється, що математичні властивості СЛАР не відповідають алгоритму розв'язування, то автоматично здійснюється перехід на алгоритми, які реалізують розв'язування іншими методами, або обчислювальний процес продовжується тим же алгоритмом з підвищеною розрядністю у відповідності до виявлених властивостей (S_7, S_8).

Вершина результат (S_9) – по закінченню розв'язування користувач отримує протокол з детальним поясненням обчислювального процесу та розв'язок задачі з оцінками достовірності.

4.4 Програмна реалізація InparSolver

Функціонально InparSolver є програмний засіб, що реалізує автоматичне виконання на комп'ютері таких етапів робіт: формування задачі з наближеними

даними на мові предметної області; автоматичне дослідження її математичних властивостей та побудова відповідного адаптивного алгоритму розв’язання; планування паралельних обчислень на ефективній моделі розпаралелення гібридного комп’ютера; формування ефективної топології комп’ютера та синтез відповідної паралельної програми; розв’язання задачі з оцінками достовірності отриманого розв’язку; пояснення процесу дослідження та розв’язування задачі, візуалізація результатів розв’язування на мові предметної області. Крім того, користувач може застосовувати необхідні адаптивні з InparSolver (як reuse-програми) для створення прикладного програмного забезпечення розв’язування науково-технічних задач з різних предметних областей, що зводяться до розв’язування СЛАР.

Всі складові InparSolver реалізовані як окремі компоненти, що дозволяє легко адаптувати програмний засіб до обчислювального середовища користувача. Користувачі мають ідентичнийний діалоговий інтерфейс для різних архітектур паралельних комп’ютерів, віддаленого та локального доступу та різних операційних систем (Linux, Windows), що дозволяє їм легко підключати потрібні програми та вивантажувати непотрібні. Це економить комп’ютерні ресурси.

На СКІТ-4 InparSolver функціонує в операційному середовищі Linux. За основу програмної платформи взято мову Python. Python - це високорівнева, інтерпретована мова програмування, яка відзначається простотою синтаксису та великою ефективністю. Ось деякі основні характеристики Python:

Інтерпретована мова: Python використовує інтерпретатор, що означає, що вам не потрібно компілювати код перед його виконанням. Це полегшує розробку та відладку коду.

Високорівнева мова: Python надає високорівневі структури даних та вбудовані функції, що дозволяють зручно та швидко розв'язувати різноманітні завдання.

Об'єктно-орієнтоване програмування: Python підтримує об'єктно-орієнтований підхід до програмування, що дозволяє створювати модульний та легко розширюваний код.

Широка бібліотека: Python має велику стандартну бібліотеку, яка включає в себе різні модулі та пакети для різних завдань, таких як обробка тексту, мережеве програмування, обробка зображень і багато іншого.

Платформенна незалежність: Python підтримує платформенну незалежність, що означає, що програми, написані на Python, можуть запускатися на різних операційних системах, таких як Windows, Linux та macOS.

Динамічна типізація: Python використовує динамічну типізацію, що дозволяє змінювати типи змінних без явного вказання типу.

Спільнота та підтримка: Python має активну та велику спільноту користувачів та розробників. Це сприяє обміну ідеями, розвитку сторонніх бібліотек та рішень, а також надає можливість швидко отримати допомогу та відповіді на питання.

Для реалізації веб інтерфейсу використано веб фреймворк Django. Django - це відкритий веб-фреймворк, написаний на мові програмування Python, призначений для швидкої розробки веб-застосунків. Він пропонує високорівневий інтерфейс для створення веб-застосунків і включає в себе багато вбудованих функцій, що полегшують роботу розробників.

Основні особливості Django включають:

- ORM (Object-Relational Mapping): Django має вбудовану ORM, яка дозволяє вам взаємодіяти з базою даних, використовуючи об'єктно-орієнтований код замість SQL-запитів.
- Адміністративний інтерфейс: Django надає готовий адміністративний інтерфейс, який автоматично генерується на основі моделей даних вашого додатка, що полегшує управління даними.
- URL-роутинг: Django використовує концепцію URL-роутингу, яка дозволяє вам визначати, як URL-адреси повинні відповідати різним частинам вашого коду.
- Шаблони: Django використовує систему шаблонів для відокремлення логіки від представлення веб-сторінок.
- Система аутентифікації та авторизації: Django має вбудовану систему для управління користувачами, аутентифікацією та авторизацією.
- Middleware: Django використовує middleware для обробки запитів та відповідей на різних рівнях.

Для контролю за задачами які можуть виконуватись на кластері довгий період часу використовується Celery. Celery - це розподілена система обробки завдань (task queue) для мови програмування Python. Головна ідея Celery полягає в тому, щоб відокремити обчислювальні завдання від основного додатку та виконувати їх асинхронно. Основні компоненти та концепції Celery:

1. Message Broker: Celery використовує брокер повідомлень для передачі та зберігання завдань. Популярні брокери повідомлень, які

використовуються з Celery, включають RabbitMQ, Redis, та інші. В нашій реалізації використовується Redis.

2. Task: Це фрагменти коду, які ви хочете виконати асинхронно. Вони визначаються у вигляді функцій або класів, і Celery відповідає за їхню виконавчу частину.
3. Worker: Це процес, який виконує завдання, що надсилаються до Celery. Багато workerів може бути запущено паралельно, щоб обробляти завдання асинхронно.
4. Celery Beat: Це компонент, який відповідає за планування та періодичне виконання завдань у фоновому режимі.
5. Configuration Management: Celery надає можливість налаштовувати різні параметри, такі як брокер повідомлень, з'єднання з базою даних, часові обмеження та інші.

Для підтримки роботи бази даних встановлено та налаштовано MySQL сервер. Для встановлення зв’язку з кластером використовується бібліотека ssh2-python.

4.5 Математичне моделювання фізико-механічних процесів

У таблиці 4.1 наведено результати апробації паралельного алгоритму на розв’язанні декількох СЛАР з розрідженими матрицями хмарочосного формату. При цьому для кожної із задач проводилося розв’язання з матрицею вихідної (неоптимізованої) структури та переупорядкованої (оптимізованої) за

допомогою алгоритму мінімального степеня – у таблиці ці випадки відрізняються значеннями ширини стрічки та заповнення профілю матриць.

Таблиця 4.1 – Час (хвилини) виконання обчислень для різної кількості потоків

Порядок	Ширина стрічки	Наповненість ненульовими елементами	1	8	16	32	64	128
44 436	4 475	21%	0.3	0.0621	0.029	0.016	0.026	0.033
44 436	37 580	2%	0.26	0.05	0.0275	0.012	0.018	0.087
283 031	19 530	7%	32	5.95	2.77	1.3	1.013	1.45
283 031	281 341	1%	5.38	0.83	0.54	0.28	0.18	0.35
661 590	34 242	5%	98.8	17.6	10.08	5.47	3.45	5.22
661 590	541 257	1%	32.45	5.17	3.08	1.57	1.01	1.55

InparSolver було використано для математичного моделювання деяких процесів у практичних застосуваннях на кластерному комплексі СКІТ. Розглянемо їх.

Статичний розрахунок напружено-деформованого стану промислової споруди. Загальний вигляд конструкції та використовувана скінченно-елементна сітка подані на рис. 4.3.

Конструкцію розбито на 13876 скінченних елементів. У результаті сформовано скінченно-елементну сітку з 7563 вузлами. Після дискретизації отримано СЛАР з матрицею порядку 44436.

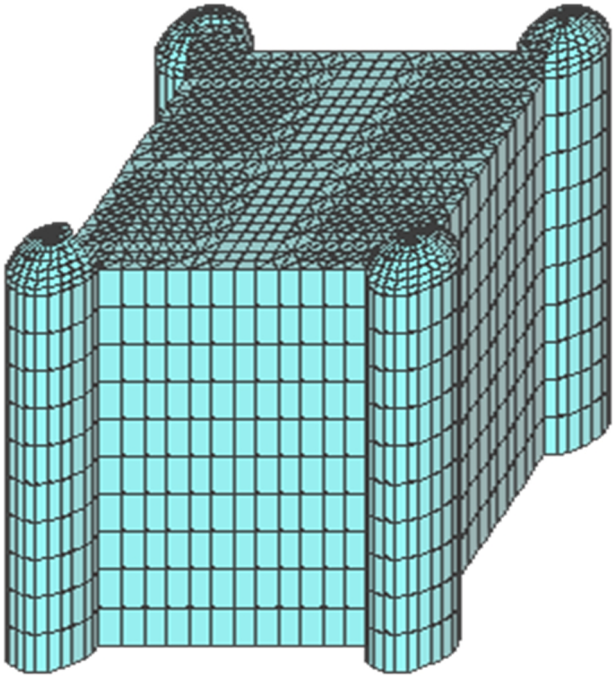

Рисунок 4.3 – Загальний вигляд конструкції

Матриця цієї системи до застосування алгоритму впорядкування має стрічкову структуру (рис. 4.4. зліва) з щільністю (заповненістю) 21%. Застосування алгоритму мінімальної степені для зміни впорядкування ненульових елементів матриці дозволила зменшити заповненість до 2%, хоча при цьому ширина півстрічки зросла (з 4476 до 27850). Структуру оптимізованої матриці подано на рис. 4.4. праворуч.

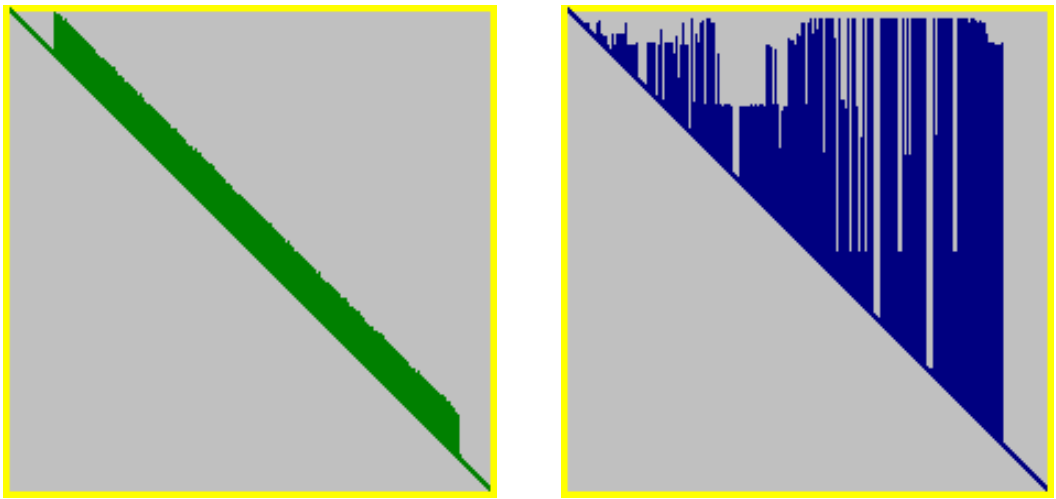

Рисунок 4.4 – Портрет матриці до та після впорядкування елементів

На рис. 4.5 показано графіки прискорення при розв’язані СЛАР з матрицями оптимізованої та оригінальної структури з використанням різної кількості потоків та розміром блоку 128. Надалі на рисунках розрахунок з матрицею вихідної структури позначений "О", а з переупорядкованою структурою – "П".

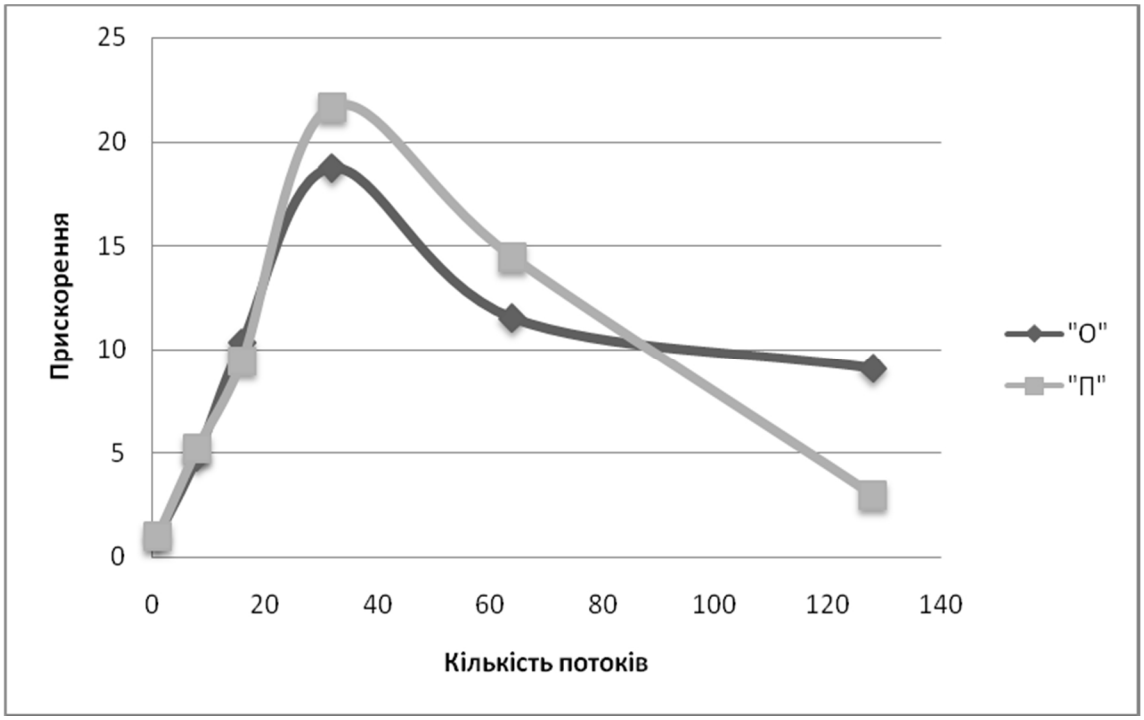

Рисунок 4.5 – Залежність прискорення від кількості потоків для СЛАР 44 436-го порядку

Статичний розрахунок напружено-деформованого стану багатоповерхового житлового будинку. Загальний вигляд конструкції багатоповерхового житлового будинку та використовувана скінченно-елементна сітка подані на рис. 4.6. зліва. Конструкція розбита на 121761 скінченних елементів. Відповідна скінченно-елементна сітка містить 110265 вузлів.

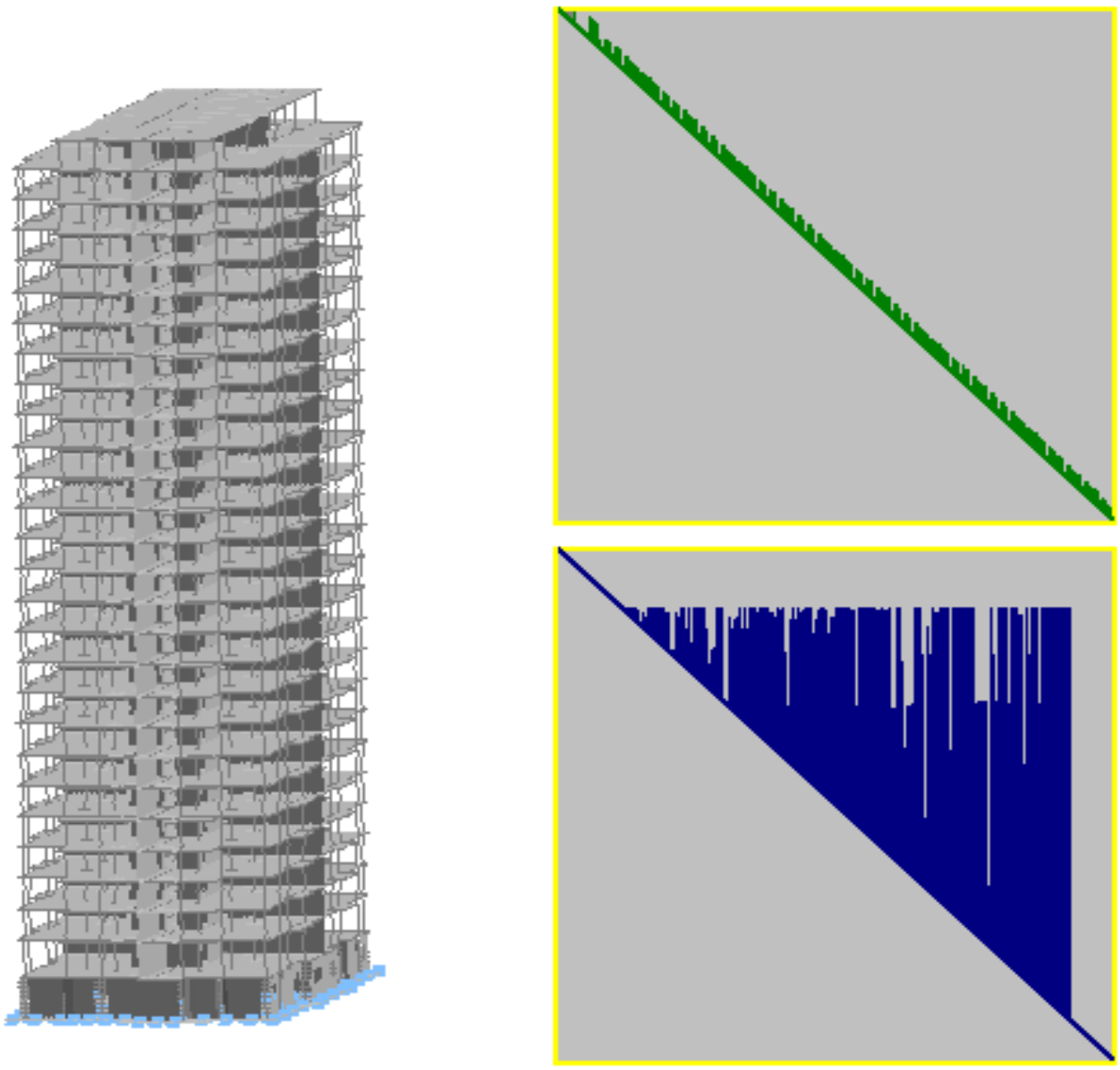

Рисунок 4.6 – Конструкція багатоповерхового житлового будинку та портрети відповідної матриці жорсткості до та після впорядкування елементів

У результаті дискретизації отримано СЛАР порядку 661590. Вихідна структура матриці цієї системи мала щільність 5% і півширину стрічки 34242 (верхня права частина рис. 4.6). Після оптимізації структури матриці щільність впорядкованої структури складала 1% при півширині стрічки, рівній 541257 (нижня права частина). На рис. 4.7 показано графіки прискорень отриманих при розв’язанні

задачі з відповідними розрідженими матрицями на різній кількості потоків з використанням розміру блоку 256.

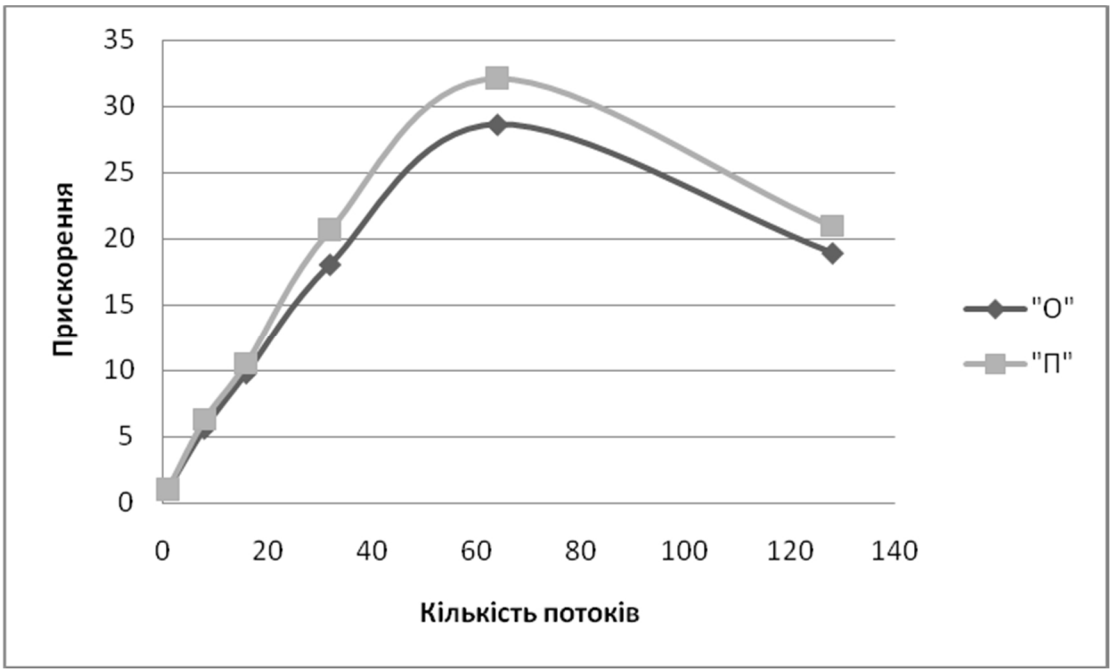

Рисунок 4.7 – Залежність прискорення від кількості потоків для СЛАР 661 590-го порядку

Аналіз напружено-деформованого стану фундаменту будівлі. Загальний вигляд конструкції та скінченно-елементна сітка подані на рис. 4.8 знизу. Конструкція розбита на 94346 скінченних елементів. Відповідна скінченно-елементна сітка містить 97412 вузлів.

У результаті дискретизації отримано СЛАР порядку 283 031. Вихідна структура матриці цієї системи мала щільність 7% і півширину стрічки 19 530 (верхня ліва частина рис. 4.8). Після оптимізації структури матриці щільність впорядкованої структури складала 1% при півширині стрічки, рівній 281 341 (верхня права частина).

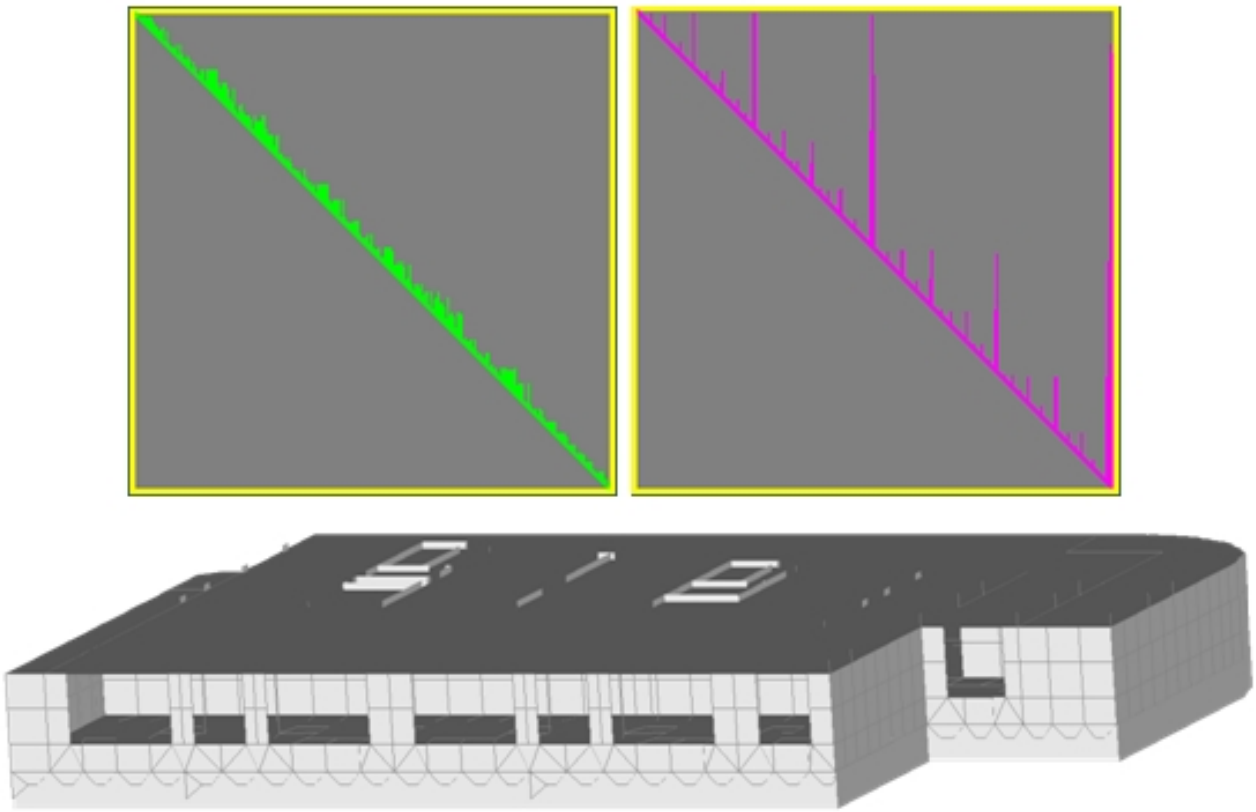

Рисунок 4.8 – Конструкція фундаменту та портрети відповідних матриць жорсткості

На рис. 4.9 показано графіки прискорень отриманих при розв’язанні задачі з відповідними розрідженими матрицями на різній кількості потоків з використанням розміру блоку 160.

Наведені на представлених рисунках результати математичного моделювання демонструють залежність характеристик алгоритму (часу розв’язання задачі, прискорення) від збалансованості завантаження обчислювальних елементів. При цьому чим менший порядок системи, тим складніше досягти збалансованості при нарощуванні числа процесів. Також проведені дослідження вказують, що значний вплив на прискорення обчислень має розмір блоку, з яким виконуються

обчислення, оскільки він впливає як на швидкість послідовних операцій так і на кількість операцій які можуть бути виконані паралельно.

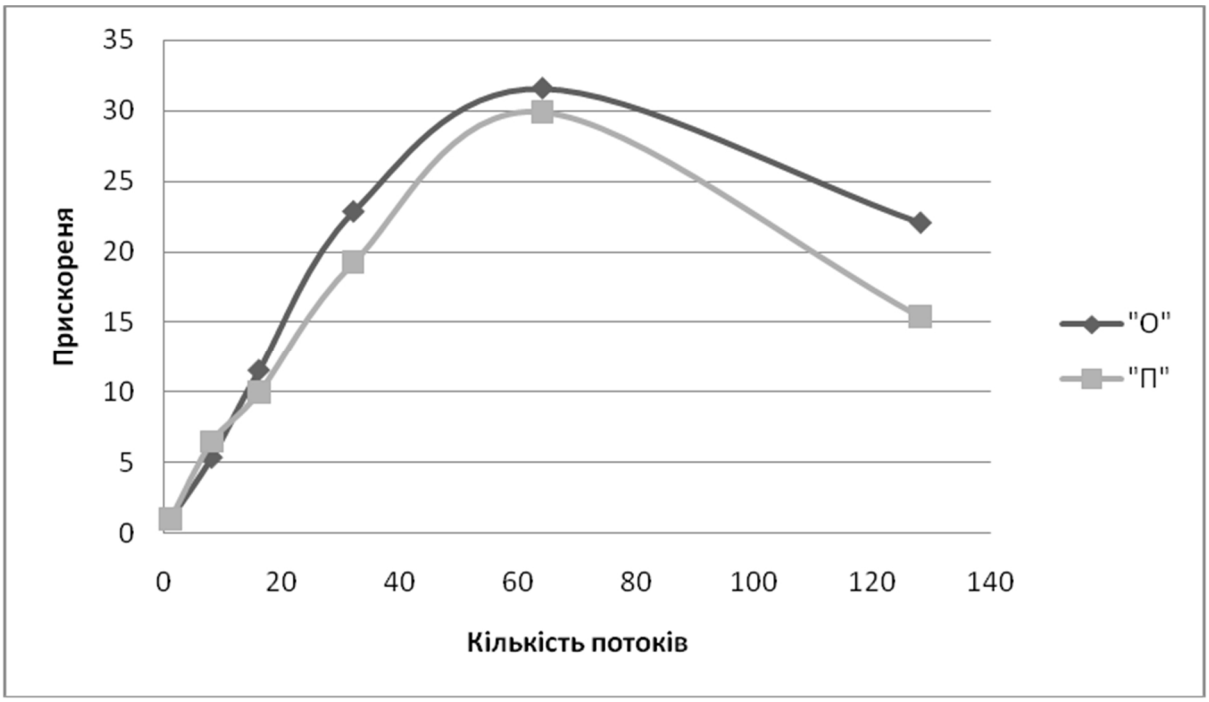

Рисунок 4.9 – Залежність прискорення від кількості потоків для СЛАР 283031-го порядка

4.6 Висновки

У розділі 4 визначено принципи створення, архітектуру та склад ІСКМ для математичного моделювання в науці і інженерії (InparSolver) для автоматичного адаптивного налаштування методу, алгоритму і топології комп'ютера на математичні властивості СЛАР з наближениим даними та структуру розріджених матриць, виявленої в результаті дослідження, та отримання комп'ютерного розв'язку з оцінкою достовірності для гібридних комп'ютерів. Розроблено алгоритми автоматичного визначення структури розріджених

матриць та їх регуляризації на основі методів машинного начання і нейромережевих технологій. Побудовано формальні моделі предметної області та користувача, на основі яких здійснено програмну реалізацію InparSolver на суперкомп'ютері СКІТ, розв'язано низку задач міцністного аналізу конструкцій в будівельний галузі.

Основні переваги InparSolver:

- постановка та дослідження СЛАР з наближеними даними;
- автоматизація дослідження та розв'язування СЛАР з наближеними даними з оцінкою достовірності комп'ютерних результатів;
- автоматизація всіх процесів розпаралелення на багатоядерному комп'ютері MIMD-архітектури з графічними процесорами при ефективному використанні комп'ютерних ресурсів;
- принцип «прихованого паралелізму», який забезпечує роботу користувача на паралельному комп'ютері, як на комп'ютері стандартної однопроцесорної архітектури.

ЗАГАЛЬНІ ВИСНОВКИ

У дисертаційній роботі розроблено інтелектуальну систему комп’ютерної математики для математичного моделювання в науці і інженерії (InparSolver) для автоматичного дослідження та розв’язування систем лінійних алгебраїчних рівнянь з матрицями різної структури та наближеними даними для комп’ютерів MIMD-архітектури з графічними процесорами, що забезпечує достовірність отримуваних результатів при ефективному використанні комп’ютерних ресурсів.

Основні результати дисертаційної роботи:

- запропоновано принципи автоматизації процесу дослідження та розув’язуваання СЛАР з розрідженою структурою даних;
- розроблено паралельний алгоритм трикутної факторизації матриць блочно-хмарочосної структури;
- розроблено метод розпізнавання структури розрідженої матриці на основі машинного навчання та згорткових нейронних мереж для вибору ефективного алгоритму розв’язування;
- запропоновано комп’ютерний алгоритм ідентифікації математичних властивостей матриць (виродженість, погана обумовленість) з використанням багаторозрядної арифметики для забезпечення достовірності комп’ютерних розв’язків;
- створено інтелектуальну систему комп’ютерної математики InparSolver для автоматизації дослідження та розв’язування СЛАР з розрідженими матрицями з функцією адаптивного налаштування методу, алгоритму, топології паралельного комп’ютера на математичні властивості задачі.

- проведено експериментальне дослідження функціональних можливостей ІСКМ InparSolver на різних моделях паралельних обчислень на практичних задачах;
- інтегровано InparSolver у математичне забезпечення суперкомп'ютера родини кластерів СКІТ.

Отримані в дисертації результати мають теоретичне та практичне значення для моделювання фізико-механічних процесів, що зводяться до розв'язування СЛАР, на сучасних високопродуктивних суперкомп'ютерах. Створена ІСКМ може використовуватися в галузях, які мають на сьогодні дуже важливе значення для оборонної промисловості і післявоєнної відбудови при створенні нових композитних матеріалів в літакобудуванні, суднобудуванні, ракетобудуванні, в дослідженнях міцності конструкцій промислового і цивільного будівництва, для математичного моделювання фізико-технічних процесів в електрозварюванні та споріднених процесах тощо.

Застосування ІСКМ надасть можливість суттєво перерозподілити роботи по постановці і розв'язанню задач між користувачем і комп'ютером у порівнянні з традиційними технологіями, скоротити терміни розроблення прикладних застосунків для розв'язання науково-технічних задач і підвищити якість одержуваних комп'ютерних рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сергієнко І.В., Хіміч О.М. Математичне моделювання: Від мелм до екзафлопсів. *Вісник НАН України*. 2019. Т. 8. С. 37–50.
2. Хіміч О.М. Суперкомп’ютерні технології та математичне моделювання складних систем. *Вісник НАН України*. 2018. Т. 5. С. 69–72.
3. Leland R.W., Rajan M., Heroux M.A., Doerfler D. Performance, Efficiency, and Effectiveness of Supercomputers. Technical Report SAND2016-3730. Albuquerque, New Mexico: Sandia National Laboratories, 2016. 20 p.
4. Kleppe A. Software Language Engineering: Creating Domain_Specific Language Using Metamodels. N.Y.: Addison-Wesley Professional, 2008. 240 p.
5. TOP 500. URL: http://www.TOP500.org/.
6. Hojjat A. Supercomputing in engineering analysis. CRC Press, 2020. 368 p.
7. Сергиенко И.В., Химич А.Н., Яковлев М.Ф. Методы получения достоверных решений систем линейных алгебраических уравнений. *Кибернетика и системный анализ*. 2011. № 1. С. 68-80.
8. Хіміч О.М., Полянко В.В., Чистякова Т.В. Паралельні алгоритми розв’язування лінійних систем на гібридних комп’ютерах. *Кібернетика та комп’ютерні технології: Зб. наук. пр.* 2020. № 2. С. 53-66
9. Dongarra J., Beckman P., Show T. all 65 authors. The International Exascale Software Project Roadmap 1. *The International Journal of High Performance Computing Applications.* 2011. Vol. 25, 1. P.3-60.
10. Химич А.Н., Молчанов И.Н., Попов А.В., Чистякова Т.В., Яковлев М.Ф. Параллельные алгоритмы решения задач вычислительной математики. Киев: Наукова думка, 2008. 247 с.

11. Сергиенко И.В., Молчанов И.Н., Химич А.Н. Интеллектуальные технологии высокопроизводительных вычислений. *Кибернетика и системный анализ*. 2010. № 5. С. 164–176.
12. Хіміч О.М., Чистякова Т.В., Баранов А.Ю. Принципи створення інтелектуального інтерфейсу для розв'язування систем лінійних рівнянь на комп'ютерах гібридної архітектури. *Проблеми програмування*. 2012, №. 2–3. С. 435–442.
13. Попов О.В., Чистяков О.В. Про ефективність алгоритмів з багаторівневим паралелізмом. *Науковий збірник «Фізико-математичне моделювання та інформаційні технології»*. 2021. Вип. 33. С. 133–137.
14. Химич А.Н., Попов А.В., Чистякова Т.В., Яковлев М.Ф. Интеллектуальна система компьютерной математики для высокопроизводительных вичислений. *Искусственный интеллект*. 2013. № 4. С. 138–149.
15. Dongarra J.J., Moler C.B., Bunch J.R., Stewart G.W. LINPACK users'guide. *Philadelphia. SIAM*. 1979. 368 p.
16. Garbow B.S., Boyle J.M., Dognarra J.J., Moler C.B. Matrix Eigensystem Routines – EISPACK Guide Extensions. Berlin: Springer, 1977. Vol. 51. 343 p.
17. Rump S.M. Algorithm for verified inclusions: Theory and practice. Reliability in Computing, R. E. Moore ed. San Diego: Academic press. 1988. P. 109–126.
18. Lefèvre V. The Generic Multiple-Precision Floating-Point Addition With Exact Rounding (as in the MPFR Library). *6th Conference on Real Numbers and Computers RNC 6.* Nov 2004, Dagstuhl, Germany. P. 135–145.
19. Robey R.W., Robey J.M., Aulwes R. In Search of Numerical Consistency in Parallel Programmimg. *Parallel Computing*. 2011. Vol. 37, № 4–5, P. 217-229.

20. Хіміч О.М., Ніколаєвська О.А., Баранов І.А. Особливості застосування багаторозрядної арифметики в математичному моделюванні. *Фізико-математичне моделювання та інформаційні технології*. 2023. Вип. 37. р.143-148.
21. Bailey D.H., Borwein J.M. High-Precision Arithmetic: Progress and Challenges. 2013, 15 p.
22. Хіміч О.М., Чистякова Т.В. Автоматичне дослідження та розв'язування лінійних систем з використанням підвищеної точності. *Матеріали Всеукраїнської науково-практичної конференції за міжнародною участю «Інформатика та системні науки (ICH-2015)*, м. Полтава, 19−21 березня 2015 р. С. 365−367.
23. IEEE Standard for Floating-Point Arithmetic. Introduced 2008-08-29. New York: Institute of Electrical and Electronics Engineers, 2008, 70 p.
24. Collange S., Defour D., Graillat S., Iakymchuk R. Reproducible and Accurate Matrix Multiplication for High-Performance Computing. *16th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN).* Wüurzburg, Germany, Book of Abstracts. 2014, September 21–26. P. 42–43.
25. Hida Y., Li S., Bailey D.H. Library for Double-Double and Quadrat-Double Arithmetic. 2007. 24 p.
26. Bailey D.H., Borwein J.M. High-precision numerical integration: Progress and challenges. *Journal of Symbolic Computation*. 2011. Vol. 46, №7. P. 741–754.
27. Nikolaevskaja E.A., Khimich A.N, Chistyakova T.V. Programming with Multiple Precision. Studies in Computational Intelligence. Berlin, Heidelberg: Springer-Verlag, 2012. Vol. 397. 233 p.
28. Voros A. Discretized Keiper / Li approach to the Riemann hypothesis, *Experimental Mathematics*. 2018. 18 p.

29. Liu J. G., Zeng Z.F. Extended generalized hyperbolic-function method and new exact solutions of the generalized hamiltonian and NNV equations by the symbolic computation. *Fundamenta Informaticae*. 2014, Vol. 132, № 4. P. 501–517.
30. Nehra V., Sehgal R. Symbolic computation of mathematical transforms and its application: A MATLAB computational project-based approach. *IUP Journal of Electrical and Electronics Engineering*. 2015. Vol. 8, №1. P. 53–76.
31. Cao Z., Hou X. A symbolic computation approach to parameterizing controller for polynomial Hamiltonian systems. *Mathematical Problems in Engineering*. 2014. 8 p.
32. Maxfield B. Essential PTC Mathcad Prime 3.0 A Guide for New and Current Users. Elsevier, 2014. 585 p.
33. Cornea M. Precision, accuracy, roun rounding, and error propagation in exascale computing. *IEEE 21st Symposium on Computer Arithmetic*, 2013, 7–10 April. Austin, TX, USA: 2013. P. 231–234.
34. Nakasato N., Daisaka H., Fukushige T., Kawai A., Makino J., Ishikawa T., Yuasa F. GRAPE-MPs: Implementation of an SIMD for quadruple/hexuple/ octuple-precision arithmetic operation on a structured ASIC and an FPGA, *IEEE 6th International Symposium*, 2012, 22 Sept. Aizu-Wakamatsu, Japan, 2012. P. 75–83.
35. Опанасенко В.М, Хіміч О.М., Лісовий О.М., Чистякова Т.В. Розв’язання задач з підвищеною точністю обчислень решения задачи наименьших квадратов. *УсиМ*. 2011, № 1. С. 9–14.
36. Ishii M., Detrey J., Gaudry P., Inomata A., Fujikawa K. Fast modular arithmetic on the Kalray MPPA-256 processor for an energy-efficient

implementation of ECM. *IEEE Transactions on Computers*. 2017. Vol. 66, № 12. P. 2019–2030.

37. Бахвалов Н.С. Численные методы. М.: Наука, 1975. Т.1. 632 с.
38. Scott J.,Tůma M. Algorithms for Sparse Linear Systems. Springer Nature, 2023. 254 p.
39. Воеводин В.В. Вычислительные основы линейной алгебры. М.: Наука, 1977.
40. Golub G.H., Van Loan C.F. Matrix Computations. 4 edition. Johns Hopkins University Press, 2013. 784 p.
41. Джордж А., Джордж Лю. Численное решение больших разреженных систем уравнений. М.: Мир, 1984. 334 с.
42. Марчук Г.И. Методы вычислительной математики. М.: Наука, 1980.
43. Молчанов И.Н. Машинные методы решения задач прикладной математики. Алгебра, приближение функций, обыкновенные дифференциальные уравнения. Київ: Наукова думка, 2007, 550 с.
44. Писанецки С. Технология разреженных матриц. М.: Мир, 1988.
45. Самарский А.А., Николаев Е.С. Методы решения сеточных уравнений. М.: Наука, 1978. 592 с.
46. Тьюарсон Р. Разреженные матрицы. М.: Мир, 1977. 172 с.
47. Уилкинсон Дж.Х., Райнш К. Справочник алгоритмов на языке АЛГОЛ. Линейная алгебра. М.: Машиностроение, 1976. 389 с.
48. Фадеев Д.К., Фадеева В.Н. Вычислительные методы линейной алгебры. М.: Физматгиз, 1963.
49. Форсайт Дж., Малькольм М., Моулер К. Машинные методы математических вычислений. М.: Мир, 1980. 279 с.
50. Saad Y. Iterative Methods for Sparse Linear Systems. PWS Publishing Company, 2000. 448 с.

51. Дьяконов В.П. Компьютерная математика. Теория и практика. М.: Нолидж. 2001. 1296 с.
52. Клименко В.П., Ляхов А.Л., Гвоздик Д.Н. Современные особенности развития систем компьютерной алгебры. *Математичні машини і системи*. 2011, № 2. С. 3–18.
53. Ploskas N., Samaras N. GPU Programming in MATLAB. Morgan Kaufmann, 2016, 318 p.
54. Ford W. Numerical Linear Algebra with Applications: Using MATLAB. Academic Press. 2014. 628 p.
55. Madenci E., Guven I. The Finite Element Method and Applications in Engineering Using ANSYS. 2nd edition. Springer, 2015. 657 p.
56. MSC Nastran – Industry Leading Multidisciplinary FEA: URL: www.mscsoftware.com/product/msc-nastran.
57. Поспелов Г.С. Системный анализ и искусственный интеллект. *Проблемы вычислительной техники*. М.: Международный центр науч. и техн. информ. 1981. С. 21–42.
58. Поспелов Д.А. Ситуационное управление: теория и практика. М.: Наука. 1986. 288 с.
59. Поспелов Д.А. Интеллектуальные интерфейсы для ЭВМ новых поколений. *Электронная вычислительная техника. Сборник статей*. М.: Радио и связь. 1989. Вып. 3. С. 4–20.
60. Jones M.T. Artificial Intelligence. A Systems Approach. Jones and Bartlett Publishers, 2008. 516 p.
61. Luger G.F. Artificial Intelligence. Structures and Strategies for Complex Problem Solving. Sixth Edition. Addison Wesley. 2009. 779 p.

62. Chein M., Mugnier M.-L. Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs. London: Springer-Verlag, 2008. 428 p.
63. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. Питер.: 2000. 384 с.
64. Russell S.J., Norvig P., Davis E. Artificial Intelligence: A Modern Approach. 4 edition. Prentice Hall, 2020, 1132 p.
65. Čojbašič Ž.M., Nikolič V.D., Čirič I.T. et al. Computationally Intelligent Modelling and Control of Fluidized Bed Combustion Process. *Thermal Science*. 2011. Vol.15, № 2. P. 321–338.
66. Simjanovska M., Gusev M., Madevska-Bogdanova A. Intelligent modelling for predicting students' final grades. *Proc. of 37th Int. Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija. 2014. P. 1216–1221.
67. Парасюк И.Н., Сергиенко И.В. Пакеты программ анализа данных: технология разработки. М.: Финансы и статистика. 1988. 159 с.
68. Молчанов И.Н., Чистякова Т.В., Химич А.Н. К вопросу создания интеллектуального Software для решения систем линейных алгебраических уравнений. *Управляющие системы и машины*. 1998. № 6. С.57–62.
69. Хіміч О.М., Попов О.В., Чистякова Т.В., Рудич О.В., Чистяков О.В. Інтелектуальна система для дослідження та розв’язування задач на власні значення на паралельних комп’ютерах з процесорами Intel Xeon Phi. *Штучний інтелект*. 2017. № 2. С. 119–127.
70. Чистякова Т.В., Єршов П.С. Про вибір розрядності обчислень в інтелектуальній системі обробки матриць. *Математичне та*

комп'ютерне моделювання. Серія: Фізико-математичні науки. 2019. Вип. 19. С. 193–198.

71. Хіміч О.М., Чистякова Т.В., Сидорук В.А., Єршов П.С. Інтелектуальна система комп'ютерної математики INPARSOLVER. *Штучний інтелект*. 2020. № 4, т.25. С. 60–71.
72. 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, №6. P. 990–997.
73. Химич А.Н. Оценки полной погрешности решения систем линейных алгебраических уравнений для матриц произвольного ранга. *Сб. науч. трудов Компьютерная математика*. Киев: Институт кибернетики им. В.М. Глушкова НАН Украины. 2002. № 2. С. 41–49.
74. Химич А.Н. Оценки возмущений для решения задачи наименьших квадратов. *Кибернетика и системный анализ*. 1996. № 3. С. 142–145.
75. Химич А.Н., Войцеховский С.А., Брусникин В.Н. О достоверности линейных математических моделей с приближенно заданными исходными данными. *Математические машины и системы*. 2004, № 3. С. 54–62.
76. Химич А.Н., Николаевская Е.А. Анализ достоверности компьютерных решений систем линейных алгебраических уравнений с приближенно заданными исходными данными. *Кибернетика и системный анализ*. 2008, Т. 44, № 6. С. 83–95.
77. Николаевская Е.А., Химич А.Н. Оценка погрешности взвешенного нормального псевдорешения с положительно-определенными весами. *ЖВМ*. 2009. Vol. 49, № 3. С. 42–430.

78. 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.
79. Попов А.В., Чистяков О.В. Про ефективність алгоритмів з багаторівневим паралелізмом. *Фізико-математичне моделювання та інформаційні технології*. 2021. Вип. 33. С. 133–137.
80. Message Passing Interface Forum. MPI: A message-passing interface standard. *International Journal of Supercomputer Applications*. 1994. Vol.8, №3/4. P. 157-416.
81. Yliluoma J. Guide into OpenMP: Easy multithreading programming for C++. http://bisqwit.iki.fi/story/howto/openmp/.
82. Soyata T. GPU parallel program development using CUDA. CRC Press, 2018. 401 p.
83. Getting started with OpenCL and GPU Computing. https://www.eriksmistad.no/getting-started-with-opencl-and-gpu-computing/.
84. 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.
85. Khimich A., Sydoruk V., Yershov P. Intellectualization Of Computation Based On Neural Networks For Mathematical Modeling. *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, 18-20 Dec. 2019. IEEE: 2019. P.445-448.
86. Buttari A, Langou J, Kurzak J, Dongarra J. A Class of Parallel Tiled Linear Algebra Algorithms for Multicore Architectures. *Parallel Computing*. 2009. Vol. 35, № 1. P. 38–53.

87. Хіміч О.М., Сидорук В.А. Використання мішаної розрядності у математичному моделюванні. *Математичне та комп'ютерне моделювання. Серія: Фізико-математичні науки. Зб. наук. праць*. 2019. Вип. 19. С. 180–187.
88. Хіміч О.М., Чистякова Т.В., Сидорук В.А., Єршов П.С. Адаптивні алгоритми дослідження задач в змінному комп'ютерному середовищі. *Фізико-математичне моделювання та інформаційні технології*. 2021. Вип. 33. С. 181–185.
89. Сидорук В.А., Єршов П.С. Адаптивний алгоритм розв'язання систем рівнянь з блочно-хмарочосними матрицями. *Міжнародний науково-технічний журнал «Проблеми керування та інформатики»*. 2022. №5. С. 17–31.
90. Math Kernel Library. URL: https://software.intel.com./en-us/mkl
91. Чистяков О.В., Ніколайчук О.О., Єршов П.С. Про математичне моделювання задач стійкості конструкцій на сучасних комп'ютерах. *Компьютерная математика*. 2018. № 2. С. 30−37.
92. Городецкий А.С., Евзеров И.Д. Компьютерные модели конструкций. К.: ФАКТ, 2007. 394 с.
93. Haykin S. Neural networks and learning machines. 3rd edition. Pearson Education, 2009. 937 p.
94. Калан Р. Основные концепции нейронных сетей. Пер. с анг. Издательский дом «Вильямс», 2001. 287 с.
95. Graupe D. Principles of artificial neural networks. 3 edition. World Scientific Publ., 2013. 363 p.
96. Ciresan D.C., Meier U., Masci J., Gambardella L.M., Schmidhuber J. Flexible, High Performance Convolutional Neural Networks for Image Classification. *International Joint Conference on Artificial Intelligence*. 2011. P. 1237–1242.

97. Python. URL: https://www.python.org/
98. Keras. URL: https://keras.io/
99. TensorFlow. URL: https://www.tensorflow.org/
100. Годунов С.К., Рябенький В.С. Разностные схемы. М.: Наука, 1973. 400 с.
101. Самарский А.А., Николаев Е.С. Методы решения сеточных уравнений. М.: Наука, 1978. 592 с.
102. Стренг Г, Фикс Дж. Теория метода конечных элементов. М.: Мир, 1977. 349 с.
103. Bathe K.J., Zimmermann P. Finite-Elemente-Methoden. Second Edition. Spriner, 2007. 634 p.
104. Химич А.Н., Молчанов И.Н, Мова В.И. и др. Численное программное обеспечение MIMD–компьютера Инпарком. Киев: Наукова думка, 2007. 222 с.