УДК 004.8

Є. О. Пружина
В.О. Денисюк

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ ВЗАЄМОДІЇ ТРЬОХ ЗАДАЧ T1, T2, T3 ЗА ДОПОМОГОЮ OPENMP

Вінницький національний технічний університет

Анотація

Розглянуто розробку програми реалізації взаємодії трьох задач. Розглянуто питання аналізу даної предметної області, поставлено задачу на виконання, обгрунтовано засоби розробки програмної частини, розроблено діаграми компонентів та діяльності програмного забезпечення, обгрунтовано вибір програмного засобу реалізації, створено програмну реалізацію та проведено тестування програмної реалізації взаємодії трьох задач. Програмна реалізація дозволяє збільшити швидкодію, продуктивність, та надійність аналізу вхідних даних. Одержані результати можливо використовувати у різноманітних алгоритмах та програмних засобах для збільшення швидкодії взаємодії задач.

Ключові слова: *паралельні обчислення, взаємодія задач, потік, секція.*

Abstract

The development of a program for implementing the interaction of three tasks is considered. The analysis of this subject area is considered, the task is set, the means of developing the software part are justified, diagrams of components and software activities are developed, the choice of software implementation is justified, software implementation is created, and testing of the software implementation of the interaction of three tasks is carried out. The software implementation allows for increased speed, performance, and reliability of the analysis of input data. The results obtained can be used in various algorithms and software tools to increase the speed of task interaction.

Keywords: *parallel computing, task interaction, thread, section.*

Вступ

Актуальність теми зумовлена швидким розвитком сучасних мов програмування, які надають широкі можливості для побудови масштабованих, модульних та кросплатформних рішень. У світовій та вітчизняній практиці спостерігається тенденція переходу від вузькоспеціалізованих до інтегрованих систем, здатних одночасно вирішувати кілька задач у рамках одного програмного комплексу. Існуючі програмні аналоги часто мають обмежену взаємодію між модулями або не забезпечують достатньої гнучкості при зміні вхідних параметрів, що підкреслює доцільність створення нового програмного продукту з покращеними характеристиками. Метою роботи є розробка програмного продукту сучасною мовою програмування, який реалізує взаємодію трьох задач T1, T2 і T3, забезпечуючи ефективний обмін даними між ними та можливість масштабування.

Постановка задачі дослідження

Задачі дослідження полягають у вирішенні наступних питань: проаналізувати існуючі підходи до реалізації взаємодії між програмними модулями; розробити архітектуру програмного комплексу, що включає три взаємопов’язані задачі; реалізувати алгоритми обчислення та обміну даними між задачами T1, T2 і T3; провести тестування розробленої програми для перевірки коректності взаємодії модулів; оцінити ефективність створеного програмного продукту.

Виклад основного матеріалу

Паралельні обчислення є основою сучасних високопродуктивних інформаційних технологій, оскільки дають змогу одночасно виконувати кілька процесів для підвищення швидкодії системи. Завдяки розвитку багатоядерних процесорів, кластерних та розподілених систем паралельні методи обчислень набули широкого застосування у вирішенні складних задач науки, техніки та

промисловості. Основна мета технологій паралелізму полягає у створенні умов, що дозволяють комп'ютерними засобами виконувати більший обсяг роботи за той самий період часу [1].

Одним із ключових понять паралельних обчислень є багатозадачність – властивість системи або мови програмування забезпечувати паралельне виконання декількох потоків чи процесів. Для коректної взаємодії потоків у таких системах використовуються механізми синхронізації: м'ютекси, семафори, умовні змінні, монітори. Сучасні мови програмування (C++, C#, Java, Python, JavaScript) надають високорівневі засоби організації багатопоточності, такі як Async/Await, Task, Executor Framework, Message Passing Interface (MPI). Застосування таких підходів підвищує ефективність, безпеку та масштабованість програмних систем, зменшує ризик виникнення блокувань та інших проблем синхронізації [2–4]. У роботі розроблено паралельну програмну систему, яка реалізує взаємодію трьох задач T1, T2, T3, організованих за принципом моделі «начальник–підлеглі».

T1 (начальник) виконує функції керування, ініціює створення підлеглих потоків, розподіляє між ними завдання, координує їхню роботу та збирає результати.

T2 (підлеглий 1) відповідає за попередню обробку даних – фільтрацію та нормалізацію.

T3 (підлеглий 2) виконує аналітичну обробку – обчислює середні значення, екстремуми, дисперсію, статистичні показники та формує підсумковий результат.

Для взаємодії між потоками використовується механізм синхронізованих черг, які забезпечують коректний обмін даними та запобігають конфліктам доступу до спільних ресурсів. Додатково застосовано атомарні змінні, що сигналізують про завершення певних етапів обчислень.

Математичне моделювання системи здійснено з урахуванням потокового обміну даними. Пропускна здатність кожного потоку визначається кількістю елементів, що обробляються за одиницю часу, а загальна пропускна здатність системи обмежується найповільнішим потоком. Прискорення паралельної обробки оцінюється за законом Амдала, а ефективність – як відношення прискорення до кількості потоків. При рівномірному балансуванні навантаження можливо досягти майже лінійного прискорення [5–7].

Для підвищення швидкодії програмної реалізації використано технологію OpenMP, яка забезпечує ефективне розподілення обчислень між ядрами процесора. Оптимізація досягалася за рахунок:

- балансування навантаження між потоками за допомогою директиви #pragma omp parallel for;
- зменшення кількості синхронізацій через використання паралельних секцій #pragma omp sections;
- оптимізації доступу до пам'яті через локальні змінні;
- можливість масштабування програми шляхом зміни параметра OMP_NUM_THREADS без зміни у коді.

Реалізація програми здійснена мовою C++, що забезпечує високу продуктивність, гнучкість і кросплатформеність. Завдяки підтримці стандарту OpenMP було створено три паралельні секції, які працюють одночасно: T1 – генерація або зчитування даних; T2 – фільтрація; T3 – аналітична обробка результатів.

Для візуалізації архітектури та логіки системи розроблено UML-діаграму компонентів, яка відображає структуру програми, черги даних. Також розроблено UML-діаграму діяльності, що демонструє схему взаємодії потоків .

Програмно реалізовано задану задачу та з використанням UML описано всі компоненти коду [8].

Аналіз результатів тестування програми виконано для різної кількості елементів масиву (табл.1).

Результати тестування підтвердили коректність алгоритму та ефективність обраного підходу. При збільшенні розміру вхідного масиву (від 10 000 до 1 000 000 елементів) програма продемонструвала стабільність і масштабованість. На чотириядерній системі отримано прискорення виконання у 3– 3,5 рази порівняно з послідовним виконанням.

Таким чином, розроблений паралельний метод взаємодії трьох задач забезпечує: підвищення швидкодії та продуктивності системи; надійну синхронізацію потоків; зменшення простоїв процесора; гнучке масштабування під апаратні ресурси.

Подальше вдосконалення може передбачати використання GPU-обчислень, розширення кількості потоків або впровадження механізмів адаптивного балансування навантаження.

Таблиця 1 – Порівняння часу виконання програми з різними вхідними даними

Розмір вхідного масиву	Час виконання першої секції, мс	Час виконання другої секції, мс	Час виконання третьої секції, мс	Загальний час виконання, мс
10000	6,6541	14,0277	18,2010	20,1250
100000	48,7056	131,7910	137,5190	140,8970
1000000	509,5750	1462,9900	1480,6840	1485,566

Висновки

Досліджено паралельний програмний комплекс, який реалізує взаємодію трьох задач T1, T2 і T3 за моделлю «начальник–підлеглі». Розроблена система демонструє ефективну організацію потокової обробки даних, забезпечує високу продуктивність і коректну синхронізацію потоків завдяки використанню технології OpenMP у середовищі програмування C++. Створено алгоритм взаємодії трьох потоків: T1 – генерація або зчитування даних; T2 – попередня фільтрація даних; T3 – аналітична обробка результатів. Взаємодія між потоками реалізована через спільні черги, м'ютекси та умовні змінні, що забезпечує надійну синхронізацію без конфліктів доступу до ресурсів. Також наведено UML-діаграми.

Програмно реалізовано поставлену задачу. Розроблена програма є універсальною, підтримує вибір користувачем режиму роботи та дозволяє відображати статистичні результати і час виконання кожної секції. Використання директив OpenMP забезпечило паралельне виконання задач без необхідності ручного керування потоками. Проведено тестування розробленої програми, Результати експериментів підтвердили правильність реалізації та стабільність роботи програми при великих обсягах даних.

Визначено, що можливими напрямами вдосконалення програми може бути: зменшення накладних витрат синхронізації, адаптація до більшої кількості процесорних ядер, реалізація підтримки GPU-обчислень, а також розширення алгоритму для обробки складніших типів даних.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Коцовський В. М. Теорія паралельних обчислень: навчальний посібник. Ужгород: ПП «АУТДОР-Шарк», 2021. 188 с.
2. Багатозадачність. URL: https://uk.wikipedia.org/wiki/Багатозадачність.
3. Герасимов В. В., Матвєєва Н. О. Розробка програмного забезпечення на платформі Java. Багатопоточне програмування і паралельні обчислення: навчальний посібник – Дніпро, 2020. – 174 с.
4. Anthony Williams. C++ Concurrency in Action. 2nd Edition. Manning Publishing, 2019. 592 p. URL: https://readingtoday.site/download/c-concurrency-in-action-second-edition
5. Parallel Scaling Guide. URL: https://rc-docs.mines.edu/pages/user_guides/Parallel_Scaling_Guide.html?utm_source=chatgpt.com.
6. Закон Амдала. URL: https://uk.wikipedia.org/wiki/Закон_Амдала.
7. Пропускна здатність. URL: https://uk.wikipedia.org/wiki/Пропускна_здатність.
8. The Unified Modeling Language. URL: https://www.uml-diagrams.org/.

Пружина Євгеній Олегович – студент кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: prugina2006dl@gmail.com;

Денисюк Валерій Олександрович – канд. техн. наук, доцент, доцент кафедри комп'ютерних наук, Вінницький національний технічний університет, м.Вінниця, e-mail: vad64@i.ua.

Pruzhyna Yevhenii Olehovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: prugina2006dl@gmail.com;

Denysiuk Valerii Olexandrovich – Ph.D., Assistant Professor, Assistant Professor of the Chair of Computer Science, Vinnytsia National Technical University, Vinnytsia, e-mail: vad64@i.ua