СИНЕРГІЯ ІНТЕРНЕТ-ТЕХНОЛОГІЙ

Матеріали

І ВСЕУКРАЇНСЬКОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ

(26 березня 2026 року)

м. Івано-Франківськ
2026 рік

Міністерство освіти і науки України
Заклад вищої освіти
«Університет Короля Данила»
Факультет суспільних і прикладних наук
Кафедра інформаційних технологій

СИНЕРГІЯ ІНТЕРНЕТ-ТЕХНОЛОГІЙ

Матеріали І Всеукраїнської науково-практичної конференції

(м. Івано-Франківськ, 26 березня 2026 року)

Івано-Франківськ

2026

DOI 10.33098/2026.1.26.03

УДК 004.774(0.064)

С 38

Рекомендовано до розміщення в електронних сервісах
ЗВО «Університет Короля Данила»
(протокол № 8 від 26 березня 2026 р.)

С 38 **Синергія інтернет-технологій**: матеріали І Всеукраїнської науково-практичної конференції (м. Івано-Франківськ, 26 березн. 2026 року). Івано-Франківськ: ЗВО «Університет Короля Данила», 2026. 156 с.

ISBN 978-617-8850-10-4

Видання вміщує тези доповідей учасників І Всеукраїнської науково-практичної конференції «Синергія інтернет-технологій», яка відбулася 26 березня 2026 року у закладі вищої освіти «Університет Короля Данила». Розраховане на наукових та науково-педагогічних працівників закладів вищої освіти і наукових установ, здобувачів вищої освіти, а також на широкий читацький загал.

Організаційний комітет не завжди поділяє думку учасників конференції. Відповідальність за достовірність фактів, статистичних даних, точність викладеного матеріалу покладається на авторів.

УДК 004.774(0.064)

4. Kumar M. RAG vs. Agentic RAG: The Evolution of Intelligent Retrieval Systems. *LinkedIn.* URL: https://www.linkedin.com/pulse/rag-vs-agentic-evolution-intelligent-retrieval-systems-maneesh-kumar-k5v2c/

5. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *arXiv.org.* URL: https://arxiv.org/abs/2404.16130

УДК 004.5

Демчина Микола,
доцент кафедри інформаційних технологій,
кандидат технічних наук,
ЗВО «Університет Короля Данила»,
м. Івано-Франківськ, Україна
ORCID: https://orcid.org/0009-0002-9161-4843

ПАРАЛЕЛІЗАЦІЯ ОБЧИСЛЕНЬ У АГЕНТНИХ СИСТЕМАХ НА ОСНОВІ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

Часто продуктивність сучасних агентних систем на основі великих мовних моделей (LLM) значною мірою обмежується їхнім послідовним характером обробки завдань. Спочатку інтелектуальний агент аналізує запит, далі використовує у своїй роботі певне джерело або інструмент, очікує на відповідь, робить проміжний висновок і лише після цього переходить до наступного кроку. Такий підхід забезпечує хорошу керованість і відтворюваність, але призводить до накопичення затримок та різко знижує швидкодію під час виконання завдань, що складаються з окремих незалежних кроків [1].

Проте під час виконання завдань, пов’язаних із зовнішніми ресурсами (вебпошук, запити до баз даних, виклик сторонніх API-сервісів), значна частина часу витрачається на очікування відповіді, а сумарний час відповіді обчислюється як сума затримок на кожному із кроків виконання. Це призводить до виникнення так званого “sequential bottleneck”. Навіть за умови ефективного процесу міркування інтелектуального агента загальна продуктивність цілісної системи може сприйматися як недостатня, якщо виконання незалежних перевірок або підзадач відбувається послідовно, тобто кожна наступна операція розпочинається лише після завершення попередньої. Такий підхід призводить до накопичення затримок і зниження ефективності використання обчислювальних ресурсів [1].

У зв’язку з цим у сучасних підходах до проєктування хмарних та агентних систем особлива увага приділяється застосуванню паралельної обробки завдань. Зокрема, вона є доцільною для вирішення задач, що передбачають оцінювання кількох альтернативних варіантів відповіді, аналізу великої кількості документів або отримання різних спеціалізованих

інтерпретацій однієї й тієї ж самої проблеми. Використання паралельних механізмів обробки на рівні інтелектуальних агентів дозволяє суттєво скоротити час, необхідний на виконання окремих завдань, та підвищити загальну ефективність інтелектуальних систем.

Одним із можливих рішень цієї проблеми є використання під час роботи інтелектуальних агентів таких шаблонів паралелізації, як “concurrent orchestration”, “fan-out/fan-in” та “scatter-gather”. Вони забезпечують механізми декомпозиції задачі на незалежні підзадачі, паралельний запуск виконання окремих завдань та агрегацію результатів у цілісну відповідь. Використання такої архітектурної схеми передбачає, що декілька інтелектуальних агентів виконуються одночасно, формуючи незалежні проміжні результати. Надалі ці результати можуть бути агреговані або узагальнені з метою отримання фінального рішення чи узгодженого висновку.

Ідея про те, що загальний час виконання окремого завдання визначається часом виконання його найдовшого підзавдання, інтуїтивно відображає концепцію критичного шляху у паралельних робочих процесах. Якщо підзадачі є незалежними, затримка отримання кінцевого результату наближається до максимальної тривалості окремих підзадач із додаванням накладних витрат, пов’язаних із координацією виконання та подальшим синтезом результатів. Це обмеження було продемонстровано у роботі Джина Амдала (Gene M. Amdahl) [2]. Внаслідок нього навіть за значного збільшення кількості агентів (виконавців) потенційне прискорення системи обмежується часткою операцій, що принципово не піддаються паралелізації.

Подальші інтерпретації закону Амдала в епоху багатоядерних обчислювальних систем дозволили виділити дві важливі закономірності в роботі агентів. По-перше, чинники, які часто залишаються поза увагою на початкових етапах проєктування, такі як послідовні фази виконання, накладні витрати та механізми координації, зі зростанням рівня паралелізму можуть набувати домінуючого впливу на продуктивність системи в цілому. По-друге, оптимізація послідовної частини обчислювального процесу в окремих випадках може мати стратегічно важливіше значення, ніж просте збільшення кількості паралельних виконавців [2]. Подібна закономірність спостерігається і в інтелектуальних агентних системах, у яких процедура синтезу результатів роботи, що включає агрегацію, перевірку коректності та узгодження суперечливих відповідей, може перетворюватися на окремий вузол критичного шляху виконання.

Таким чином, паралелізація завдань не може розглядатися як безумовне джерело підвищення швидкодії системи в цілому. Її застосування

фактично призводить до перерозподілу часових витрат. Замість очікування результатів послідовного виконання зовнішніх викликів значна частина часу та ресурсів витрачається на вирішення завдань координації, агрегації результатів та контролю їхньої якості. Найбільша ефективність від паралельного виконання досягається у випадку виконання незалежних підзадач, які мають зазвичай I/O-орієнтований характер, оскільки їх паралельний запуск дозволяє мінімізувати простої, пов'язані з очікуванням завершення операцій введення-виведення.

Під час проєктування інтелектуальних агентів перехід від послідовної до паралельної моделі виконання завдань часто реалізується за допомогою подієво-орієнтованого підходу на основі шаблону "scatter-gather". У межах цього підходу координатор здійснює розподіл підзадач між паралельними виконавцями (scatter), після чого отримані результати збираються (gather) для подальшої обробки. На етапі інтеграції результатів виконання можуть застосовуватися різні стратегії, зокрема об'єднання отриманих відповідей, їх порівняння або вибір оптимального результату. Важливо зазначити, що "scatter-gather" є передусім координаційним шаблоном організації обчислювального процесу. Він передбачає очікування результатів від кількох паралельних гілок виконання та подальшу їх агрегацію з метою формування узгодженого рішення.

Зокрема, платформа Amazon Web Services у своїх рекомендаціях щодо застосування шаблонів agentic AI описує підхід на основі "scatter-gather" як механізм паралельної обробки множини підзадач із подальшим агрегуванням отриманих результатів у консолідоване рішення. У відповідних рекомендаціях від Amazon також наведено приклади практичної реалізації такого підходу із використанням керованих сервісів паралельного виконання та механізмів кореляції результатів виконання [3].

Якщо розглядати платформу OpenAI, то тут паралелізація реалізується на двох основних рівнях. По-перше, вона може проявлятися у вигляді паралельних викликів для інструментів або функцій (tool/function calling) у межах одного агента. По-друге, застосовується підхід на основі створення спеціалізованих агентів, результати роботи яких надалі інтегруються метаагентом у межах схеми "fan-out/fan-in" [4].

Проте використання паралельних механізмів виконання істотно змінює вимоги до інженерії агентних систем. На передній план виходять питання ефективної декомпозиції завдань, узгодженої роботи зі станом системи, проєктування механізмів агрегації результатів, а також забезпечення належного рівня спостережуваності процесу виконання.

Як перспективний напрям розвитку агентних систем, паралельна оркестрація окремих агентів природним чином призводить до формування архітектур, у яких взаємодіють цілі команди агентів, що виконують різні ролі та використовують спеціалізовані механізми узгодження. У межах такої парадигми питання про те, яким чином агенти можуть ефективно співпрацювати паралельно не лише в межах одного робочого процесу, а й між кількома взаємопов'язаними робочими процесами, трансформується у самостійну дослідницьку проблему.

Зокрема, актуальними стають питання масштабування механізмів координації інтелектуальних агентів, забезпечення узгодженості результатів їх роботи та керування зростаючою складністю системи таким чином, щоб підвищення швидкодії не супроводжувалося зниженням рівня керованості та якості отримуваних результатів.

Як підсумок, паралелізація не може розглядатися лише як окремий прийом оптимізації на рівні окремих інтелектуальних агентів, а виступає фундаментальним структурним принципом проєктування агентних систем в цілому. Її застосування передбачає стратегічне визначення тих етапів обчислювального процесу, де послідовне виконання є достатнім, а також тих, де доцільною є паралельна оркестрація підзадач. Водночас ефективне використання паралелізації потребує чітко визначених інженерних підходів до декомпозиції завдань і подальшої агрегації результатів, що дозволяє забезпечити керованість, надійність та передбачуваність отримуваного прискорення в роботі інтелектуальних агентів.

Список використаних джерел:

1. Workflow for parallelization. *AWS Prescriptive Guidance*. URL: https://docs.aws.amazon.com/prescriptive-guidance/latest/agentic-ai-patterns/workflow-for-parallelization.html (дата звернення: 13.03.2026).

2. Hill M. D., Marty M. R. Amdahl's Law in the multicore era. *IEEE Computer*. URL: https://research.cs.wisc.edu/multifacet/papers/ieeecomputer08_amdahl_multicore.pdf (дата звернення: 13.03.2026).

3. Parallelization and scatter-gather patterns. *AWS Prescriptive Guidance*. URL: https://docs.aws.amazon.com/prescriptive-guidance/latest/agentic-ai-patterns/parallelization-and-scatter-gather-patterns.html (дата звернення: 13.03.2026).

4. Function calling. *OpenAI API Documentation*. URL: https://developers.openai.com/api/docs/guides/function-calling/ (дата звернення: 13.03.2026).