УДК 004.272 https://doi.org/10.15407/pp2026.02.028

*Д.Ю. Ліпатов, С.Л. Хрипко*

# АДАПТИВНА МОДЕЛЬ УЗГОДЖЕНОСТІ ТА ПАРАЛЕЛЬНОЇ ОБРОБКИ В РОЗПОДІЛЕНИХ БАЗАХ ДАНИХ

У статті запропоновано та досліджено нову модель адаптивного керування узгодженістю та паралельною обробкою в розподілених базах даних для високонавантажених інформаційних систем. Метою розробки моделі є подолання фундаментального протиріччя між необхідністю забезпечення високого рівня узгодженості даних (що досягається шляхом збільшення параметрів кворуму та синхронізації між репліками) та вимогами до низької затримки і високої пропускної здатності системи в умовах інтенсивного паралельного навантаження. Запропонована модель формалізує динамічний механізм адаптації параметрів розподіленої системи, зокрема рівнів узгодженості та ступеня паралелізму, на основі безперервного аналізу стану системи. Модель реалізує замкнений адаптивний цикл, побудована на основі мікросервісного підходу із використанням контейнеризації, що забезпечує масштабованість та гнучкість налаштування. Експериментальні результати демонструють, що модель забезпечує зниження середньої затримки обробки запитів на 20–40%, а також підвищення пропускної здатності на 15–30% за рахунок адаптивного керування паралельністю, водночас досягається стабілізація рівня узгодженості даних, що проявляється у зменшенні частоти конфліктів у разі динамічних змін навантаження. Практичне значення роботи полягає у можливості застосування запропонованої моделі для оптимізації продуктивності та надійності сучасних інформаційних систем, зокрема у сферах фінансових технологій, електронної комерції, Інтернету речей та хмарних сервісів.
Ключові слова: розподілені бази даних, адаптивна узгодженість, паралельна обробка, контейнеризація, високонавантажені системи, оптимізація продуктивності

*D.Yu. Lipatov, S.L. Khrypko*

# AN ADAPTIVE CONSISTENCY AND PARALLELISM MODEL IN DISTRIBUTED DATABASES

This paper proposes and investigates a novel model for adaptive management of consistency and parallel processing in distributed databases for high-load information systems. The goal of the model is to overcome the fundamental trade-off between ensuring a high level of data consistency (achieved through increased quorum parameters and synchronization among replicas) and the requirements for low latency and high system throughput under intensive parallel workloads.

The proposed model formalizes a dynamic mechanism for adapting key parameters of a distributed system, in particular consistency levels and the degree of parallelism, based on continuous monitoring and analysis of the system state. The model implements a closed-loop adaptive control cycle and is designed using a microservices architecture with containerization, which ensures scalability and flexibility of configuration.

Experimental results demonstrate that the model reduces the average request processing latency by 20–40% and increases throughput by 15–30% through adaptive parallelism control, while maintaining a stable level of data consistency, reflected in a reduced conflict rate under dynamically changing workloads.

The proposed approach has practical significance for optimizing the performance and reliability of modern information systems, particularly in domains such as financial technologies, e-commerce, the Internet of Things, and cloud services.
Keywords: distributed databases, adaptive consistency, parallel processing, containerization, high-load systems, performance optimization

## Вступ

Швидкий розвиток розподілених баз даних та технологій паралельних обчислень дозволив створити високонавантажені інформаційні системи, здатні обробляти великомасштабні потоки даних у режимі реального часу [6, 7, 14]. Такі системи широко використовуються в хмарних обчисленнях, платформах Інтернету речей, фінансових послугах та аналітиці в режимі реального часу, де критично важливими є як продуктивність, так і надійність [9].



 

Однак розробники систем стикаються з фундаментальною архітектурною дилемою: як забезпечити високий рівень узгодженості даних, водночас зберігаючи низьку затримку та високу пропускну здатність [2, 3, 11]. Підвищення рівня узгодженості гарантує правильність даних, але призводить до більшої затримки та зниження швидкості реагування системи [1, 11]. І навпаки, - послаблення обмежень узгодженості покращує продуктивність та масштабованість системи, але створює ризики конфліктів даних та тимчасової неузгодженості [1, 10].

Цей компроміс особливо очевидний у сучасних розподілених NoSQL-системах, таких як Dynamo та Cassandra, які надають пріоритет доступності та масштабованості за допомогою моделей остаточної узгодженості [4, 5]. Хоча такі підходи покращують відмовостійкість, вони вимагають додаткових механізмів для обробки конфліктів та забезпечення прийнятних рівнів узгодженості.

Існуючі рішення зазвичай спираються на статичні конфігурації системних параметрів та рівні паралельної обробки. Ці конфігурації можуть добре працювати за певних робочих навантажень, але стають неефективними в динамічних та гетерогенних умовах [12, 15]. Більше того, практичні дослідження розподілених програм баз даних підтверджують мінливість поведінки системи за реальних робочих навантажень, підкреслюючи необхідність адаптивних механізмів [16].

Тому існує очевидна потреба в динамічному, контекстно залежному підході, який розглядає управління узгодженістю та паралельну обробку як багатоцільову задачу оптимізації в реальному часі.

Метою даної роботи є розробка формальної моделі та практичної архітектури для адаптивного керування узгодженістю та паралельною обробкою в розподілених базах даних, орієнтованої на динамічне врахування контексту виконання. Наукова новизна полягає у формалізації контекстно залежного механізму ухвалення рішень на основі багатокритеріальної функції вартості, що враховує ключові метрики якості обслуговування (QoS) та використання ресурсів, а також у його інтеграції в контейнеризовану мікросервісну архітектуру розподіленої системи.

Гіпотеза дослідження: Адаптивна модель керування параметрами розподіленої бази даних, яка динамічно змінює рівень узгодженості та ступінь паралелізму залежно від поточного стану системи та характеристик навантаження, дозволить суттєво підвищити продуктивність, зменшити затримки обробки запитів та забезпечити контрольований рівень узгодженості даних порівняно зі статичними підходами.

## 1. Огляд проблеми та існуючих підходів

Традиційні розподілені системи спираються на сильні моделі узгодженості, реалізовані за допомогою консенсусних протоколів, що забезпечує правильність ціною зниження продуктивності [3, 13]. На противагу цьому, сучасні NoSQL-системи використовують розслаблені моделі узгодженості для досягнення кращої масштабованості та доступності [4, 5].

Паралельні фреймворки для обробки даних, такі як MapReduce та Apache Spark, дозволяють ефективно обробляти великомасштабні дані, розподіляючи обчислення між кількома вузлами [7, 14]. Однак збільшення кількості паралельних потоків не завжди приводить до покращення продуктивності через конкуренцію за ресурси та накладні витрати на синхронізацію [15].

Нещодавні дослідження розглядали адаптивні методи оптимізації розподілених систем з урахуванням робочого навантаження [12]. Крім того, механізми вирішення конфліктів, такі як безконфліктно реплікувані типи даних (CRDT), забезпечують теоретичну основу для управління узгодженістю в розподілених середовищах [10].

Незважаючи на ці досягнення, існуючі підходи часто розглядають узгодженість та паралелізм окремо, не маючи єдиного адаптивного фреймворку, який би спільно оптимізував обидва аспекти.

Це підкреслює важливість не лише високої продуктивності системи, а й забезпечення узгодженості даних, відмовостійкості та передбачуваності поведінки в умовах змінного навантаження, що є критичним викликом для розподілених систем, де параметри обробки та рівень узгодженості можуть динамічно змінюватися.

Отже, очевидною є потреба у динамічній, контекстно обумовленій моделі, яка розглядає керування узгодженістю та паралельною обробкою не як статично задану конфігурацію, а як задачу багатокритеріальної оптимізації в реальному часі. Така модель повинна враховувати не лише поточний рівень навантаження, а й сукупність параметрів системи: прогнозовану затримку обробки запитів, доступні обчислювальні ресурси, інтенсивність конфліктів між репліками, характеристики мережевої взаємодії та допустимі компроміси між узгодженістю даних і продуктивністю для конкретного сценарію використання.

## 2. Формальна модель адаптивного керування узгодженістю та паралельною обробкою

Запропонована модель реалізує підхід адаптивного керування параметрами розподіленої бази даних із динамічним вибором рівня узгодженості та ступеня паралелізму на основі спеціалізованого модуля — Adaptive Consistency and Parallelism Manager (ACPM).

Для кожного інтервалу часу або запиту Q модуль ACPM формує рішення

$$D = (R, W, T)$$

де R і W — параметри для операцій читання та запису відповідно, а T — кількість паралельних потоків обробки. Це рішення визначає режим функціонування системи залежно від поточного стану середовища та вимог до якості сервісу.

Ухвалення рішення базується на контекстному векторі:

$$C = \{C_{lat}, C_{thr}, C_{conf}, C_{res}, C_{load}\}$$

де: $C_{lat}$ — поточний рівень затримки (latency), $C_{thr}$ — пропускна здатність (throughput), $C_{conf}$ — частота конфліктів або неузгодженостей, $C_{res}$ — використання обчислювальних ресурсів (CPU, пам'ять), $C_{load}$ — характеристика навантаження (інтенсивність, read/write).

Вибір оптимального рішення формалізується як задача мінімізації функції вартості: $C_{ost}(D) = w_{lat} \cdot L(D) + w_{thr} \cdot \frac{1}{Th(D)} + w_{conf} \cdot \mathrm{Confl(D)} + w_{res} \cdot \mathrm{Res(D)}$, де: $\mathrm{L(D)} \approx \alpha(\mathrm{R, W}) + \beta(\mathrm{T})^{-1}$ — прогнозована затримка, Th(D) =T/(L + витрати) — пропускна здатність, $Confl(D) = exp(-k \cdot (R + W))$ — рівень конфліктів, Res(D) — використання ресурсів, $w_i$ — вагові коефіцієнти, що визначають пріоритети оптимізації. Вагові коефіцієнти адаптуються динамічно залежно від контексту системи. Зокрема, у разі високого навантаження зростає вагомість ($w_{thr}$), за умови зростання конфліктів — вагомість узгодженості ($w_{conf}$), а у випадку перевантаження ресурсів зростає вагомість обмеження використання ресурсів ($w_{res}$). Ваги динамічно коригуються залежно від контексту системи, що забезпечує адаптивну поведінку [11, 12]. Обробка конфліктів підтримується за допомогою механізмів на основі CRDT, що забезпечує остаточну узгодженість без глобальної синхронізації [10].

Для забезпечення коректності роботи системи вводиться обмеження узгодженості: $R + W > N$, де N — кількість реплік у кластері. У загальному випадку ACPM оцінює функцію вартості для множини допустимих конфігурацій та обирає рішення з мінімальними очікуваними витратами. Адаптивний режим у розподілених базах даних реалізується як гібридна стратегія керування, що поєднує переваги різних рівнів узгодженості та ступенів паралелізму. Такий підхід дозволяє одночасно досягати низької затримки обробки запитів, характерної для слабкої узгодженості, та високої надійності даних, притаманної строгим моделям узгодженості. Гібридний режим може бути реалізований як у паралельному варіанті, коли операції виконуються з різними рівнями узгодже-

ності з подальшою синхронізацією результатів, так і у послідовному режимі, де первинна обробка здійснюється з мінімальними витратами часу, а подальша верифікація або уточнення — із підвищеним рівнем узгодженості. Запропонований підхід забезпечує адаптивний баланс між ключовими характеристиками системи, зокрема затримкою, пропускною здатністю, рівнем узгодженості та ефективністю використання обчислювальних ресурсів. Для ефективного прогнозування якості обслуговування системи за різних конфігурацій (R, W, T) ми використовуємо модель на основі регресії, яка зіставляє спостережуване робоче навантаження та системні метрики з очікуваними порушеннями затримки, пропускної здатності та узгодженості. Це дозволяє проактивно адаптувати параметри керування.

## 3. Архітектурна реалізація моделі

Запропонована модель ACPM інтегрована в багаторівневу архітектуру розподіленої системи (Рис. 1), що включає клієнтський рівень, сервісний рівень та рівень даних. Такий підхід забезпечує модульність, масштабованість і можливість динамічної адаптації параметрів системи в умовах змінного навантаження. На клієнтському рівні, реалізованому через API Gateway, відбувається ініціація запитів Q та їх маршрутизація до розподіленого кластера бази даних. Клієнтський рівень також може включати механізми кешування та попередньої обробки запитів, що дозволяє зменшити навантаження на систему та скоротити час відгуку. На сервісному рівні ключовим компонентом є менеджер ACPM, який реалізує логіку адаптивного керування. Він взаємодіє з модулем моніторингу, що збирає інформацію про стан системи (затримка, пропускна здатність, рівень конфліктів, використання ресурсів), та формує контекстний вектор C. На основі цього вектора ACPM виконує оцінку множини можливих конфігурацій D=(R,W,T) та обирає оптимальне рішення шляхом мінімізації функції вартості. Модуль ухвалення рішень реалізовано як окремий контейнеризований

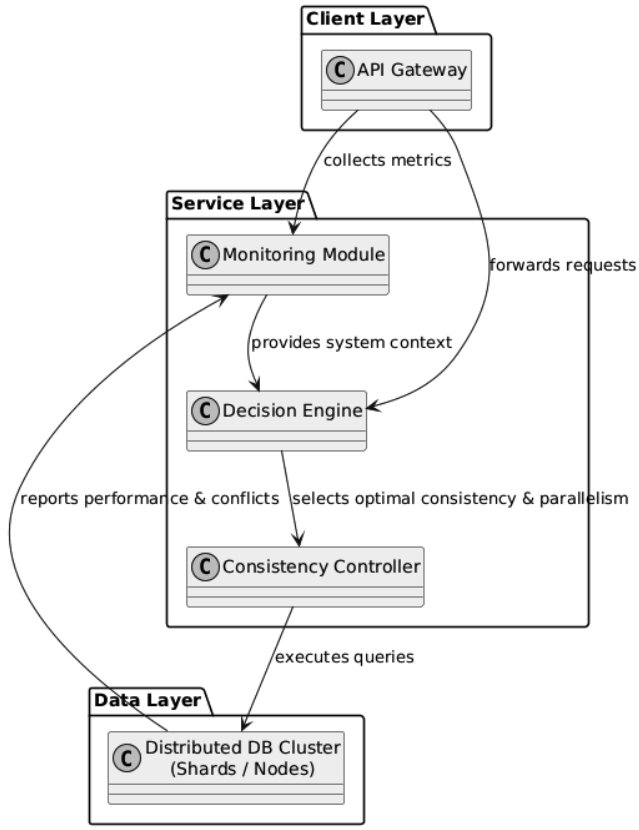


Рис. 1. Багаторівнева архітектура ACPM

сервіс, який може використовувати як евристичні правила, так і моделі машинного навчання для прогнозування поведінки системи. Модуль управління узгодженістю (Consistency Controller) відповідає за застосування обраної конфігурації, зокрема зміну параметрів (читання або запис) та налаштування рівнів узгодженості для операцій читання і запису. Паралельно виконується керування кількістю потоків обробки запитів, що дозволяє оптимізувати використання обчислювальних ресурсів. На рівні даних розгорнуто розподілений кластер бази даних (NoSQL-система), який підтримує горизонтальне масштабування, реплікацію та налаштовувану узгодженість. [4, 6] Вузли кластера функціонують у контейнеризованому середовищі, що забезпечує гнучкість розгортання та можливість швидкого масштабування. Взаємодія між вузлами відбувається через внутрішні протоколи синхронізації, що забезпечують підтримку заданого рівня узгодженості.

На відміну від традиційних механізмів автомасштабування, наш підхід включає обмеження узгодженості, гарантуючи, що рішення щодо масштабування не порушують необхідний
рівень цілісності даних.

Архітектура базується на мікросервісному підході з використанням контейнерів Docker та оркестрації. [8, 13] Основні компоненти: Кластер розподіленої БД, Модуль моніторингу, Модуль ухвалення рішень, Модуль управління узгодженістю, API-шлюз, Генератор навантаження / клієнт.

На рис.2 представлено основний алгоритм ухвалення рішення щодо вибору конфігурації обробки запиту. Процес ініціюється надходженням вхідного запиту Q, після чого виконується оцінка контексту системи, що включає аналіз рівня затримки, пропускної здатності, частоти конфліктів, завантаженості ресурсів та типу операції (читання або запис). На основі цих параметрів формується множина допустимих конфігурацій D=(R,W,T), для яких обчислюється функція вартості.

Подальший етап передбачає вибір оптимальної конфігурації шляхом мінімізації функції вартості, що відображає компроміс між продуктивністю та узгодженістю. [2, 3] У разі необхідності система може обрати гібридний режим, за якого частина операцій виконується з нижчим рівнем узгодженості для зменшення затримки, тоді як критичні операції обробляються з підвищеними вимогами до цілісності даних.

Менеджер ACPM формує контекстний вектор C на основі даних, отриманих від модуля моніторингу, який збирає інформацію про стан системи, зокрема затримку, пропускну здатність, рівень конфліктів та використання ресурсів. Далі для кож-

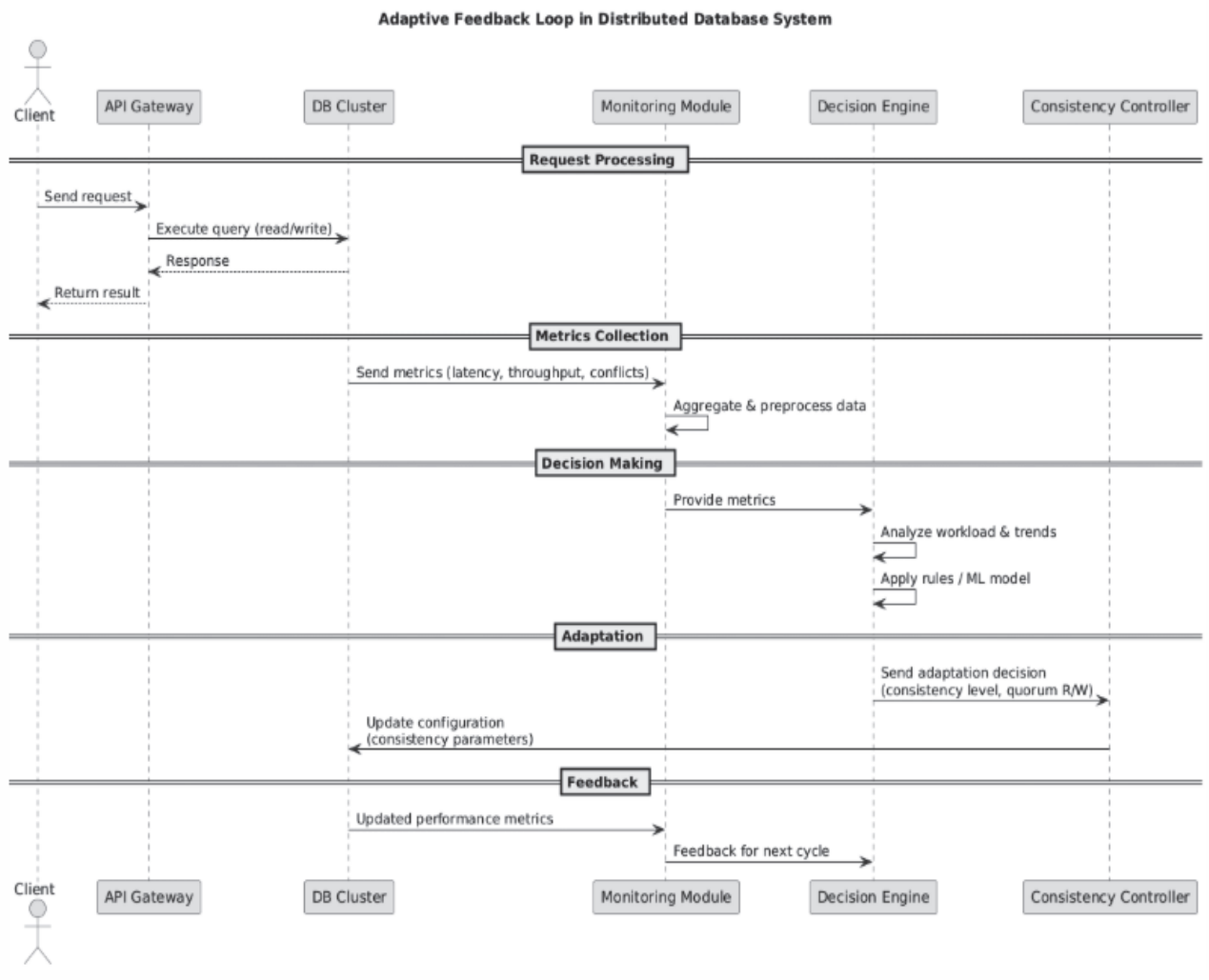


Рис. 2. Діаграма послідовності одного циклу адаптації

ної з можливих конфігурацій D=(R,W,T), що визначають рівень узгодженості та ступінь паралелізму, обчислюється значення багатокритеріальної функції вартості Cost(D). Вагові коефіцієнти $w_i$ у функції вартості адаптуються динамічно залежно від значень контексту C. Зокрема, у разі зростання затримки збільшується вагомість параметра ($w_{lat}$), за умови підвищення частоти конфліктів — вагомість узгодженості ($w_{conf}$), а під час перевантаження обчислювальних ресурсів — вагомість використання ресурсів ($w_{res}$). Такий підхід забезпечує контекстно орієнтоване керування системою в реальному часі.

Остаточне рішення D ухвалюється шляхом порівняння значень функції вартості для множини допустимих конфігурацій із урахуванням набору евристичних правил. Наприклад, у разі перевищення порогового значення затримки система автоматично знижує параметри для мінімізації затримки, тоді як при високому рівні конфліктів перевага надається конфігураціям із підвищеним рівнем узгодженості. У разі перевантаження системи додатково застосовується обмеження кількості паралельних потоків для запобігання деградації продуктивності.

На відміну від підходів CPQ-ситем, наша модель одночасно оптимізує ступінь паралелізму (T) разом з R та W, що дозволяє більш повну адаптацію до динамічних робочих навантажень. Також ми не лише оптимізуємо паралелізм а й підбираємо функцію продуктивності за допомогою регресії, щоб передбачити результати QoS. На відміну від автомасштабування, наш підхід передбачає оптимальні конфігурації за обмежень узгодженості, а не реагує лише на використання ресурсів.

## 4. Експериментальна оцінка ефективності моделі

Запропонована модель ACPM була валідована у веб орієнтований застосунок для автоматизації процесів конфігурації, ціноутворення та формування комерційних пропозицій в середовищі розподіленої інформаційної системи з високим навантаженням, виконує сценарії обробки великої кількості запитів до NoSQL-бази даних. Для оцінки ефективності адаптивного керування узгодженістю та паралельною обробкою було проведено серію експериментів, спрямованих на вимірювання ключових метрик продуктивності та порівняння з базовими підходами.

У межах першого експерименту оцінювалися характеристики системи за умови різних рівнів узгодженості. Для конфігурації зі зниженим затримки та низької узгодженості було отримано середню затримку L(low) = 70±20 мс у разі високої пропускної здатності Q(low) ≈ 1800 операцій/с, проте зі збільшеним рівнем конфліктів C(low) ≈ 0.08. Для конфігурації з підвищеним рівнем узгодженості затримка зросла до L(high) = 210±90 мс, а пропускна здатність знизилась до Q(high) ≈ 950 операцій/с, водночас рівень конфліктів зменшився до C(high) ≈ 0.01. Отримані результати підтверджують наявність класичного компромісу між затримкою та узгодженістю.

У другому експерименті було досліджено вплив ступеня паралелізму на продуктивність системи. Зі збільшенням кількості потоків обробки до T≈32 спостерігалося зростання пропускної здатності до Qmax ≈ 2000 операцій/с. Проте подальше збільшення кількості потоків призводило до ефекту насичення та незначного зниження продуктивності через конкуренцію за ресурси та накладні витрати синхронізації. Водночас середня затримка зростала з L ≈ 80 мс до L ≈ 140 мс при максимальному навантаженні.

У третьому експерименті було проведено порівняння запропонованої адаптивної моделі ACPM з базовими стратегіями: Static-High-Consistency (фіксована висока узгодженість), Static-Low-Consistency (фіксована мінімальна узгодженість), Static-Parallelism (фіксована кількість потоків). Було змодельовано 10 000 запитів із різними профілями навантаження (read-heavy, write-heavy, змішані сценарії). Результати показали, що стратегія Static-High-Consistency забезпечує мінімальний рівень конфліктів, але має найвищу затримку та

найнижчу пропускну здатність. Натомість Static-Low-Consistency демонструє найкращу продуктивність, але супроводжується високою частотою неузгодженостей. Фіксований паралелізм не дозволяє ефективно адаптуватися до змін навантаження. Запропонована модель ACPM продемонструвала середню затримку на рівні L(adaptive) ≈ 110 мс, що на 20–40% менше порівняно зі стратегією високої узгодженості, водночас пропускна здатність зросла до Q(adaptive) ≈ 1600–1900 операцій/с. Рівень конфліктів залишався контрольованим і не перевищував C(adaptive) ≈ 0.02–0.03, що є компромісним значенням між двома крайніми підходами.

Четвертий експеримент був спрямований на аналіз поведінки моделі ACPM у різних умовах навантаження. Було встановлено, що під час стабільного навантаження система обирає конфігурації з підвищеним рівнем узгодженості, забезпечуючи надійність даних. У разі різкого зростання інтенсивності запитів модель автоматично знижує параметри узгодженості та збільшує кількість потоків, що дозволяє уникнути різкого зростання затримки. У разі перевантаження обчислювальних ресурсів відбувається зменшення рівня паралелізму для стабілізації системи.

Отримані результати підтверджують ефективність запропонованої адаптивної моделі та її здатність динамічно балансувати між продуктивністю, узгодженістю та використанням ресурсів. Графічні залежності наведені на рис. 3–5, наочно демонструють нелінійний характер цих взаємозв'язків і підтверджують доцільність використання адаптивного підходу.

Апроксимація експериментальних даних виконана експоненційною моделлю, що дозволило отримати аналітичну залежність між затримкою та рівнем узгодженості. Лінеаризація через логарифмічне перетворення підтвердила адекватність експоненційної моделі та забезпечила високу точність апроксимації. Отримана функція може бути використана в алгоритмах адаптивного керування параметрами розподіленої системи.

Модель апроксимації:

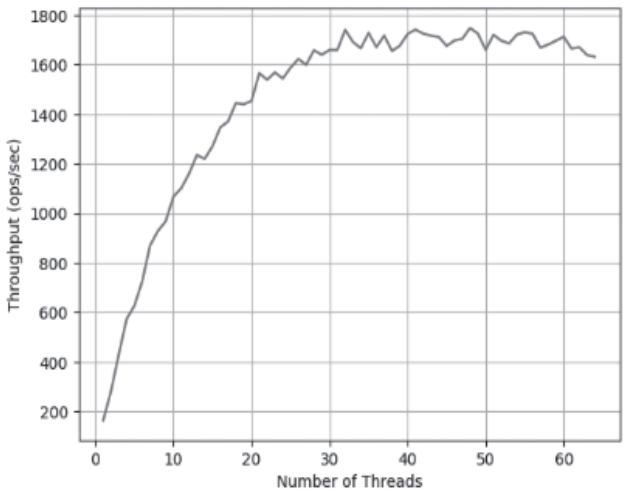


Рис. 3. Графік залежності затримки від узгодженості в моделі ACPM

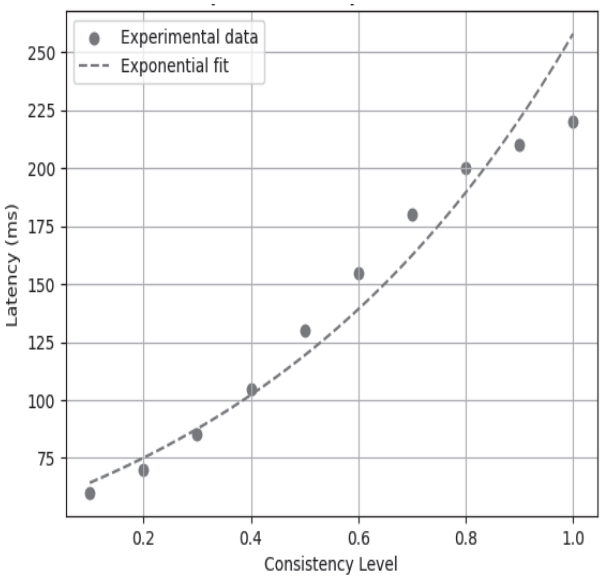


Рис. 4. Графік залежності пропускної здатності від кількості потоків в моделі ACPM

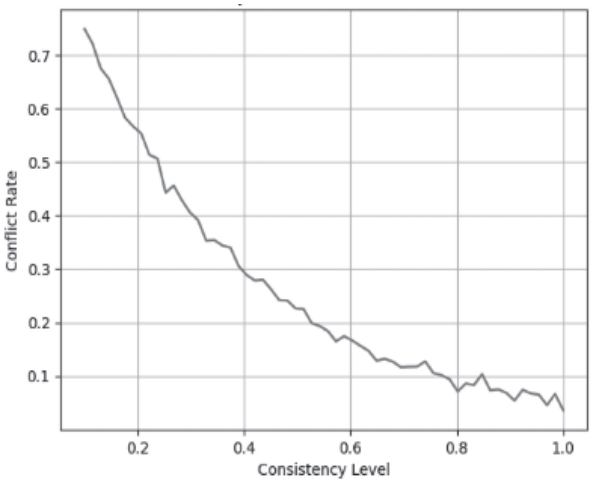


Рис. 5. Графік залежності узгодженості від конфліктів в моделі ACPM

$$L(C) = a \cdot e^{bC} + c$$

де: L(C) — затримка, C — узгодженість, a, b, c — параметри регресії.

В нашому випадку ми отримали L(C) ≈ 55,13 * exp(1,54 * C). коефіцієнт детермінації $R^2 \approx 0.97$ Отримана залежність пропускної здатності від потоків відображає поведінку, характерну для розподілених систем: Фаза 1 (1 - 20 потоків): майже експоненційне зростання пропускної здатності за рахунок ефективного паралелізму. Фаза 2 (20-40 потоків). Відбувається уповільнення росту через конкуренцію за ресурси та блокування (locks, I/O contention). Фаза 3 (>40 потоків): насичення та часткова деградація продуктивності це типовий ефект для систем типу NoSQL-кластерів.

Залежність послідовності від конфліктів має експоненційний характер спадання, так за низької узгодженості накладається високий рівень конфліктів та нестабільність даних, а у разі високої узгодженості спостерігається те, що конфлікти майже зникають, але ціною зростання затримки.

Отримані результати демонструють, що оптимальна кількість потоків існує і залежить від стану системи. При цьому оптимальне рішення не є сталим, а змінюється залежно від поточного стану системи, що обґрунтовує необхідність використання адаптивних механізмів керування.

## Висновки

У статті запропоновано та реалізовано модель адаптивного керування узгодженістю та паралельною обробкою в розподілених базах даних, орієнтовану на динамічне врахування стану системи та характеристик навантаження. Проведені експерименти підтвердили основну гіпотезу дослідження, згідно з якою контекстно залежна адаптація параметрів узгодженості та рівня паралелізму дозволяє досягти суттєво кращого балансу між затримкою, пропускною здатністю, рівнем узгодженості та ефективністю використання ресурсів порівняно зі статичними підходами.

У межах роботи формалізовано модель ухвалення рішень на основі мінімізації багатокритеріальної функції вартості, яка враховує ключові параметри системи, зокрема, затримка, пропускна здатність, рівень конфліктів і завантаженість ресурсів. Запропоновану модель (ACPM) реалізовано у веб-орієнтований застосунок для автоматизації процесів конфігурації, ціноутворення та формування комерційних пропозицій в середовищі розподіленої інформаційної системи з високим навантаженням, інтегрованого в мікросервісну архітектуру з використанням контейнеризації, що забезпечує масштабованість, гнучкість розгортання та можливість роботи в умовах змінного навантаження.

Експериментальна оцінка продемонструвала ефективність підходу: середня затримка обробки запитів була зменшена на 20–40% порівняно зі стратегіями з фіксованим високим рівнем узгодженості, а пропускна здатність зросла на 15–30% завдяки адаптивному керуванню паралелізмом. Водночас забезпечено контрольований рівень конфліктів, що не перевищує 2–3%, свідчить про досягнення збалансованого компромісу між продуктивністю та надійністю[11].

Отримані результати підтверджують доцільність використання адаптивних підходів у розподілених системах обробки даних та демонструють їхній потенціал для застосування у високонавантажених інформаційних системах, зокрема, у фінансових сервісах, електронній комерції та IoT-платформах. Перспективи подальших досліджень полягають у розробці інтелектуальних методів автоматичного налаштування вагових коефіцієнтів функції вартості, застосуванні методів машинного навчання для прогнозування поведінки системи, а також розширенні моделі на гетерогенні розподілені середовища та мультихмарні інфраструктури.

## References

1. Abadi, D. (2012) ‘Consistency tradeoffs in modern distributed database system design’, IEEE Computer, 45(2), pp. 37–42. doi:10.1109/MC.2012.33.
2. Brewer, E.A. (2000) ‘Towards robust distributed systems’, in Proceedings of the Annual ACM Symposium on Principles of Distributed Computing (PODC), pp. 7–10.

3. Cardellini, V., Colajanni, M. and Yu, P.S. (2013) ‘Adaptive load balancing in distributed systems’, IEEE Transactions on Parallel and Distributed Systems, 24(2), pp. 238–247. doi:10.1109/TPDS.2012.89.
4. Chang, F. et al. (2008) ‘Bigtable: A distributed storage system for structured data’, ACM Transactions on Computer Systems, 26(2), Article 4. doi:10.1145/1365815.1365816.
5. Curino, C. et al. (2011) ‘Workload-aware database monitoring and adaptation’, in Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 851–862. doi:10.1145/1989323.1989414.
6. Dean, J. and Ghemawat, S. (2008) ‘MapReduce: Simplified data processing on large clusters’, Communications of the ACM, 51(1), pp. 107–113. doi:10.1145/1327452.1327492.
7. DeCandia, G. et al. (2007) ‘Dynamo: Amazon’s highly available key-value store’, in Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP), pp. 205–220. doi:10.1145/1294261.1294281.
8. Gilbert, S. and Lynch, N. (2002) ‘Brewer’s conjecture and the feasibility of consistent, available, partition-tolerant web services’, ACM SIGACT News, 33(2), pp. 51–59. doi:10.1145/564585.564601.
9. Khrypko, S.L., Shcherbakov, S.S. Characteristics of additional distributed databases, in Reforming the economy in the context of international cooperation: mechanisms and strategies: proceedings of the international scientific and practical conference (Zaporizhzhja, 6–7 March 2024). Lviv–Torun: Likha-Press, pp. 74–77. doi:10.36059/978-966-397-364-7-14.
10. Kleppmann, M. (2017) Designing Data-Intensive Applications. Sebastopol: O’Reilly Media.
11. Lakshman, A. and Malik, P. (2010) ‘Cassandra: A decentralized structured storage system’, ACM SIGOPS Operating Systems Review, 44(2), pp. 35–40. doi:10.1145/1773912.1773922.
12. Shapiro, M. et al. (2011) ‘Conflict-free replicated data types (CRDTs)’, in Stabilization, Safety, and Security of Distributed Systems, pp. 386–400. doi:10.1007/978-3-642-24550-3_29.
13. Tanenbaum, A.S. and van Steen, M. (2017) Distributed Systems: Principles and Paradigms. 3rd edn. Pearson.
14. Verma, A. et al. (2015) ‘Large-scale cluster management at Google with Borg’, in Proceedings of the European Conference on Computer Systems (EuroSys), pp. 1–17. doi:10.1145/2741948.2741964.
15. Vogels, W. (2009) ‘Eventually consistent’, Communications of the ACM, 52(1), pp. 40–44. doi:10.1145/1435417.1435432.
16. Zaharia, M. et al. (2016) ‘Apache Spark: A unified engine for big data processing’, Communications of the ACM, 59(11), pp. 56–65. doi:10.1145/2934664.

Дата першого надходження до видання: 30.03.2026
Внутрішня рецензія отримана: 16.04.2026
Зовнішня рецензія отримана: 21.04.2026
Дата прийняття статті до друку: 05.06.2026
Дата публікації: 29.06.2026

***Про авторів:***

*Ліпатов Денис Юрійович,*
аспірант
*Lipatov Denys,*
post-graduate student
http://orcid.org/ 0000-0002-9665-9376

*Хрипко Сергій Леонідович,*
доктор технічних наук, професор
*Khrypko Sergiy*,
Ph.D. (technical sciences), professor
http://orcid.org/0000-0002-0647-9935

***Місце роботи авторів:***

Класичний приватний університет
Classical Private University
69002, м.Запоріжжя,
вул. Жуковського, 70б.
Тел.: (0612) 280 778