УДК 004.65 **doi:** https://doi.org/10.20998/2522-9052.2021.3.09

А. С. Карпенко[1], О. М. Тарасюк[2], А. В. Горбенко[1, 3]

[1] Національний аерокосмічний університет імені М. Є. Жуковського «ХАІ», Харків, Україна
[2] Одеський технологічний університет «ШАГ», Одеса, Україна
[3] Leeds Beckett University, Лідс, Великобританія

ДОСЛІДЖЕННЯ УЗГОДЖЕНОСТІ ТА ПРОДУКТИВНОСТІ У НЕРЕЛЯЦІЙНИХ РЕПЛІКОВАНИХ БАЗАХ ДАНИХ

Анотація. Метою даної статі є дослідження продуктивності розподілених відмовостійких інформаційних систем та нереляційних сховищ даних, а також аналізу впливу параметрів узгодженості даних на швидкодію та пропускну здатність на прикладі трьох-реплікованого кластеру Cassandra. **Результати.** У статті наведено результати навантажувального тестування (бенчмаркінгу) продуктивності операцій читання та запису кластера Cassandra, репліки якого було розгорнуто на ресурсах хмарного провайдера Amazon Web Services. Представлені кількісні результати показують, як різні налаштування узгодженості впливають на продуктивність Cassandra під час різних робочих навантаженнях в умовах, коли всі репліки розташовані в одному центрі обробки даних (ЦОД), або ж географічно розподілені по різним ЦОД. **Висновок.** Запропоновано метод мінімізації часових затримок Cassandra при гарантуванні строгої узгодженості даних на основі оптимізації налаштувань узгодженості в залежності від поточного робочого навантаження та пропорції між операціями читання та запису.

Ключові слова: NoSQL; база даних Cassandra; теорема CAP; компроміс; узгодженість; затримки; продуктивність; порівняльний аналіз.

Вступ

На сьогоднішній день, бази даних NoSQL (Non SQL або Not only SQL) [1] широко застосовуються при побудові розподілених комп'ютерних систем, включаючи соціальні мережі та різноманітні Інтернет-додатки. Сховища даних NoSQL, призначені для забезпечення горизонтального масштабування, також пропонуються хмарними провайдерами у якості послуги. Концепція нереляційних баз даних NoSQL була запропонована для ефективного зберігання та забезпечення швидкого доступу до великих обсягів інформації, т.зв. Big Data, що неможливо досягти використовуючи традиційні системи управління реляційними базами даних. Більшість NoSQL сховищ жертвують гарантіями ACID (Atomicity, Consistency, Isolation, Durability) на користь властивостей BASE (Basically Available, Soft state, Eventually consistent) [2], що є вимушеною платою за можливість розподіленої обробки даних та горизонтального масштабування.

У цій статті досліджується можливість досягнення компромісу між узгодженістю даних, доступністю системи та часовими затримками. Хоча наявність якісного взаємозв'язку між цими характеристиками була зазначена у теоремі CAP (Consistency-Availability-Partition tolerance) [3], встановлення кількісних відношень між ними є ключовим для ефективного використання NoSQL рішень. Незважаючи на бурхливий розвиток ринку нереляційних сховищ даних, галузеві тенденції свідчать про те, що Apache Cassandra входить до трійки лідерів цього ринку разом із MongoDB та HBase [4]. Оцінка та порівняння продуктивності різних баз даних NoSQL є актуальним напрямом наукових та практичних досліджень. З цією метою більшість з наявних робіт за цим напрямком, зокрема [5-8], використовують засоби навантажувального випробування (наприклад, Yahoo! Cloud Serving Benchmark, YCSB [5]). Опубліковані результати показують, що залежно від сценарію використання, умов розгортання, використовуваного навантаження та налаштувань бази даних, будь-яка база даних NoSQL може перевершити інші за певними показниками. Однак треба відмітити [9-11], що існуючі наукові доробки як правило не враховують взаємозалежність між узгодженістю інформації та продуктивністю реплікованих нереляційних баз даних, а також не досліджують, як параметри узгодженості впливають на затримку виконання операцій читання та запису.

Метою даної роботи є дослідження продуктивності, а також встановлення кількісного взаємозв'язку між узгодженістю інформації та швидкодією бази даних NoSQL, використовуючи Cassandra, в якості типового прикладу розподілених нереляційних сховищ даних з реплікацією. Apache Cassandra пропонує набір унікальних функцій (наприклад, можливість налаштування рівня узгодженості, надзвичайно швидкий запис інформації, можливість роботи в географічно розподілених центрах обробки даних, тощо) та забезпечує високу доступність без єдиної точки відмови, що робить її однією з найбільш гнучких і популярних NoSQL рішень. У статті ми також пропонуємо підхід до оптимального вибору параметрів узгодженості операцій читання та запису для забезпечення максимальної швидкодії гарантуючи строгу узгодженість даних при змішаних робочих навантаженнях.

1. Модель компромісів між узгодженістю, доступністю та швидкодією розподілених баз даних

Реплікація даних на декількох обчислювальних вузлах є традиційним методом підвищення продуктивності та надійності в розподілених інформаційних системах та активно використовуються в нереляційних базах даних. Проте, наявність багатьох реплік ускладнює процес забезпечення узгодженості даних, а також підвищує ймовірність розподілення системи (тобто втрати зв'язку з однією або кількома репліками) особливо за умов глобального розподілу реплік.

Теорема CAP, яка вперше з'явилася в 1998-1999 рр., визначає взаємозв'язок між узгодженістю даних (Consistency), доступністю системи (Availability) та стійкістю до розподілення (Partition tolerance), стверджуючи, що в розподілених реплікованих інформаційних системах одночасно можуть виконуватися лише дві з цих трьох властивостей [3]. В роботі [4] розглядають теорема CAP розглядається, як окремий випадок більш загальної системи компромісів між узгодженістю, доступністю та часовими затримками (швидкодією) в розподілених інформаційних системах.

Дійсно, ступінь реплікації (кількість наявних реплік даних), доступність системи, узгодженість даних та затримка обслуговування (час відгуку) тісно пов'язані. Більше того, ми вважаємо, що ці властивості потрібно розглядати як безперервні, а не бінарні величини. Доступність системи можна інтерпретувати, як ймовірність того, що кожен запит клієнта врешті отримує відповідь. Відсутність відповіді від деяких реплік протягом зазначеного тайм-ауту спричиняє розділення реплікованої системи. Повільний канал передачі інформації, перевантаження будь якого компоненту системи (тобто репліки) або неоптимальні налаштування часу очікування можуть призвести до помилкового рішення про те, що система стала розділеною. Коли система виявляє факт розділення, вона повинна вирішити, чи повертати клієнту потенційно неузгоджену інформацію, або ж надіслати у відповідь повідомлення про помилку або ж виняткову ситуацію, яке порушує доступність системи. Узгодженість – це також континуум, який варіюється від слабкої узгодженості (weak consistency) в одній крайності до сильної узгодженості (strong consistency) в іншій, з різними точками умовної узгодженості, або узгодженості в кінцевому підсумку (eventual consistency) між ними.

Повністю запобігти розділенню системи, яке трапляється внаслідок перевантаження середовища передачі, втрати повідомлень, хакерські атаки або збої та відмови компонентів неможливо, а отже, необхідно обирати між доступністю системи та узгодженістю інформації.

Розробники сучасних розподілених інформаційних систем та веб-додатків, таких як Facebook, Twitter тощо, часто вирішують послабити вимоги щодо узгодженості, запроваджуючи асинхронні оновлення даних, щоб досягти більшої доступності системи та забезпечити більш швидку реакцію. Проте найбільш раціональним підходом є динамічне балансування цих властивостей.

База даних Cassandra NoSQL пропонує гнучкий механізм налаштування рівня узгодженості для досягнення компромісу між доступністю, швидкодією та достовірністю.

Рівень узгодженості між даними, що зберігаються на різних вузлах-репліках можна контролювати для кожної окремої операції читання або оновлення інформації. Рівень узгодженості при виконанні операції читання даних визначає скільки вузлів-реплік має відповісти на запит перед поверненням даних клієнту. Найбільш актуальна версія даних обирається на основі порівняння часовою мітки відповіді, отриманої від кожної репліки.

У свою чергу, рівень узгодженості при виконанні операцій запису або оновлення визначає кількість реплік, які повинні підтвердити успішність виконання операції запису/оновлення (треба відзначити, що технічно немає ніякої різниці між операціями запису та оновлення). Усі операції читання та запису/оновлення Cassandra підтримують наступні основні параметри узгодженості:

- ONE: дані повинні бути записані в журнал транзакцій принаймні одного вузла-репліки, перш ніж успішність операції запису буде підтверджена клієнту; під час читання даних Cassandra запитує та повертає відповідь з однієї репліки (найближчої репліки з найменшою затримкою мережі); рівень узгодженості ONE забезпечує найбільшу швидкодію бази даних, але ж не гарантує, що клієнт отримає найбільш актуальну копію інформації;
- TWO: дані повинні бути записані щонайменше на два вузли-репліки перед підтвердженням; операція читання запрошує відповідь з двох найближчих реплік (найсвіжіші дані визначаються порівнянням міток часу записів, повернутих цими двома репліками);
- THREE: схожа на TWO, але для трьох реплік;
- QUORUM: кворум вузлів повинен підтвердити запис інформації або повернути відповідь на запит на читання; кворум обчислюється округленням до цілого числа наступної оцінки: *загальна_кількість_реплік*/2 + 1;
- ALL: дані повинні бути записані на всі вузли реплік у кластері перед підтвердженням; операція читання гарантує строгу узгодженість даних тобто гарантує повернення найбільш актуальної версії даних за рахунок очікування та порівнянні відповідей від усіх реплік; операція читання не вдасться, навіть якщо одна репліка не відповідає; вочевидь рівень узгодженості ALL буде мати найгіршу швидкодію.

Якщо репліки Cassandra розподілені по декількох центрах обробки даних (ЦОД), клієнту також будуть доступні кілька додаткових рівнів узгодженості, що визначають узгодженість даних для кожного ЦОД:

- EACH_QUORUM;
- LOCAL_QUORUM;
- LOCAL_ONE.

Якщо узгодженість даних є головним пріоритетом та необхідно забезпечити, щоб всі операції читання завжди повертали найновіші оновлення, необхідно, щоб сума вузлів які використовуються в операціях читання та запису перевищувала коефіцієнт реплікації, тобто загальну кількість реплік. Це завжди забезпечує строгу узгодженість даних. В іншому випадку можлива тільки імовірнісна узгодженість. Наприклад, строга узгодженість забезпечується, якщо:

- рівень узгодженості QUORUM встановлюється як для запитів запису, так і для читання;
- рівень узгодженості ONE встановлюється для запису та ALL для читання;
- рівень узгодженості ALL встановлюється для запису та ONE для читання.

Чим слабший рівень узгодженості, тим швидше Cassandra виконує обробку операцій читання та запису. За рахунок перерозподілу кількості вузлів, що використовуються для виконання читання та запису інформації користувачі Cassandra можуть надавати пріоритет продуктивності для одного або іншого типу операцій, все ще гарантуючи строгу узгодженість даних.

2. Навантажувальне тестування продуктивності Cassandra

У цьому та наступному розділах наведено методологія оцінки продуктивності NoSQL бази даних Cassandra та надано отримані експериментальні результати, які показують, як налаштування узгодженості впливають на час обслуговування запитів читання та запису. В якості тестового середовища ми розгорнули 3-реплікований кластер Cassandra у публічному хмарному провайдері Amazon Web Services. Коефіцієнт реплікації рівний 3 є найбільш типовою установкою для багатьох сучасних розподілених обчислювальних систем та Інтернет-додатків, включаючи Amazon S3, Amazon EMR, Facebook Haystack, DynamoDB тощо. Було досліджено два найбільш поширених варіанту розгортання кластеру Cassandra: *централізований* (рис. 1) та *розподілений* (рис. 2).

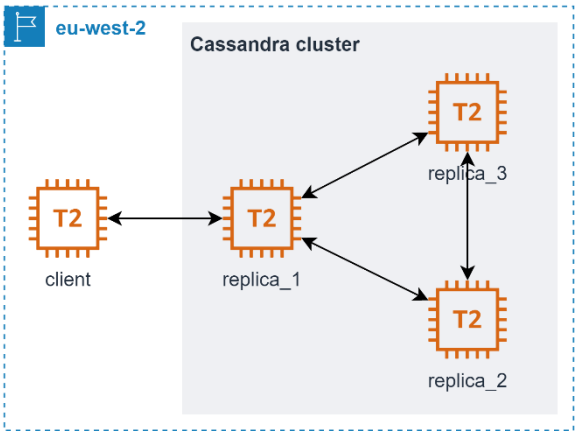


Рис. 1. Централізований кластер Cassandra
(**Fig. 1.** Cassandra centralized cluster)

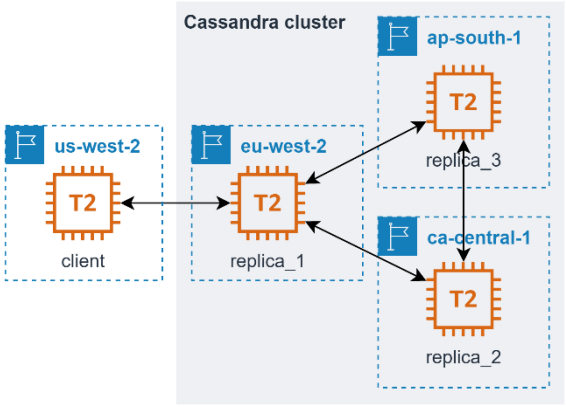


Рис. 2. Розподілений кластер Cassandra
(**Fig. 2.** Cassandra distributed cluster)

У першому випадку всі три репліки, а також клієнтська станція яка використовувалась для генерації клієнтських запитів були розташовані в одному ЦОД, а саме eu-west2. Такий варіант організації кластеру Cassandra характерний для побудови корпоративних систем збору та обробки великих даних. У випадку розподіленого кластеру клієнтська станція, а також всі три репліки кластера було розгорнуто у різних географічних регіонах Amazon: us-west-2, eu-west2, ca-central, ap-south-1. Такий варіант моделює роботу глобально-розподіленої інформаційної системи, клієнти якої також знаходяться у різних куточках мережі Інтернет, що характерно, наприклад, для соціальних мереж, або ж систем з високими вимогами до катастрофостійкості.

В усіх випадках використовувалися екземпляри віртуальних машин t2.xlarge (vCPU – 4, оперативна пам'ять – 16 ГБ, SSD – 2x50 ГБ, ОС – Ubuntu Server 16.04 LTS). Для навантажувального тестування продуктивності кластеру Cassandra було застосовано фреймворк YCSB (Yahoo! Cloud Serving Benchmark), який вважається еталоном для оцінки продуктивності хмарних інформаційних систем та баз даних NoSQL, таких як Cassandra, MongoDB, Redis, HBase та інші. YCSB - це проект Java з відкритим кодом. Структура YCSB включає шість готових робочих навантажень, кожен з яких тестує різні загальноприйняті сценарії використання з певної суміщу операцій читання та запису наприклад, 50/50, 95/5, тощо.

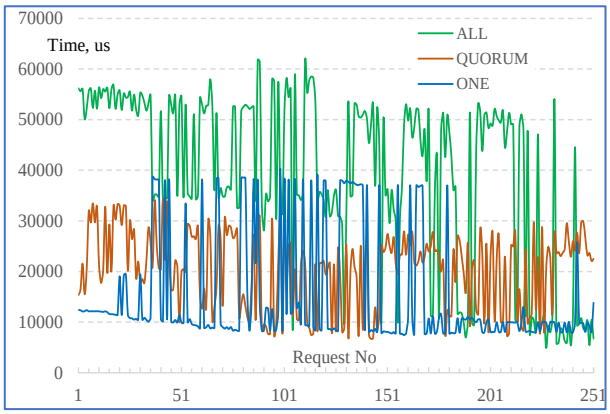
Перед початком тестування було створено та заповнено тестову базу даних YCSB, яка є таблицею записів. Кожен запис ідентифікується первинним ключем і включає F-рядкові поля. Значення, записані в ці поля, є випадковими рядками ASCII довжиною L байт. За замовчуванням F дорівнює 10, а L дорівнює 100, що створює 1000 байтові записи.

Клієнт YCSB – це програма Java, яка генерує дані для завантаження в базу даних, а потім виконує тестування її продуктивності шляхом генерації багатьох клієнтських запитів. Деякі приклади загальних методологій тестування Cassandra та інших баз даних NoSQL з YCSB можна знайти в [12]. Однак, на відміну від цих та інших робіт (наприклад, [5, 6, 7, 8]), основну увагу було зосереджено на аналізі динамічних аспектів продуктивності Cassandra при використанні різних параметрів узгодженості. Тобто, було досліджено яким чином встановлений рівень узгодженості інформації впливає на час обслуговування запитів читання та запису та пропускної здатності бази даних в залежності від рівня навантаження (тобто кількості одночасних клієнтських запитів/потоків).

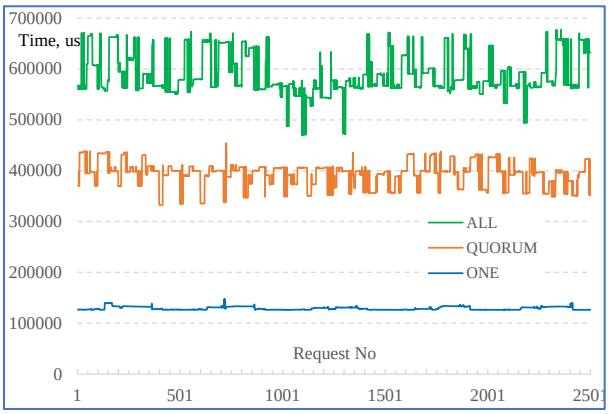
Для цього було виконано серію тестів продуктивності з кількістю потоків від 100 до 1000. Кількість операцій у кожному потоці дорівнювала 10000. Такий сценарій тестування був використаний для виконання окремо операцій читання та запису з трьома різними параметрами узгодженості: ONE, QUORUM та ALL.

Результати кожного тесту було збережено в окремому файлі формату csv, включає наступну інформацію для кожної виконаної операції читання та запису: тип операції (READ | WRITE), час початку операції (мс), затримку обслуговування операції (нс). Це дозволило побудувати графіки затримки обслуговування (див. приклад на рис. 3 та 4 (кожен графік

накладає три криві, що відповідають різним параметрам узгодженості: ONE, ALL, QUORUM), а також усереднити час обслуговування операцій читання та запису (рис. 4, 5).

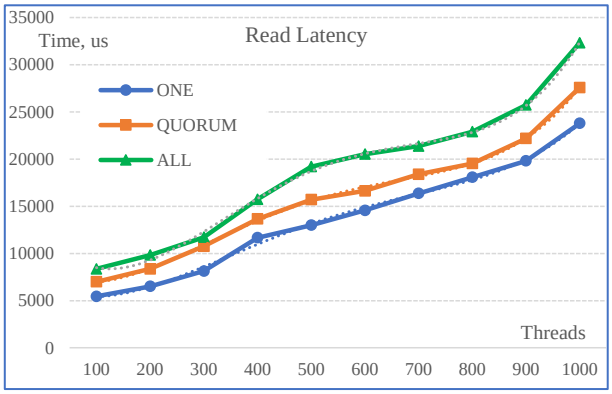


а – централізований кластер

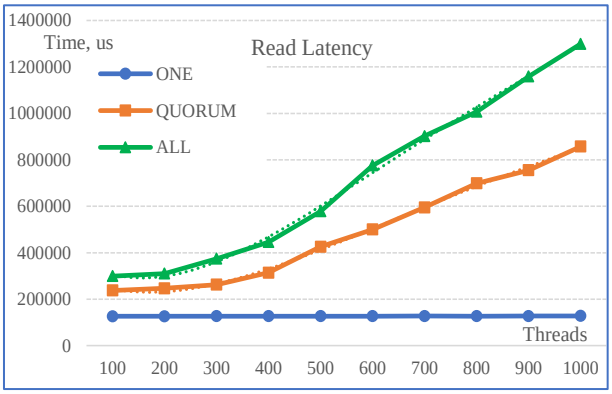


б – розподілений кластер

Рис. 3. Фрагмент графіку зміни часових затримок для операцій читання, кількість потоків = 500
(**Fig. 3.** Fragment of the graph of changes in time delays for reading operations, the number of threads = 500)

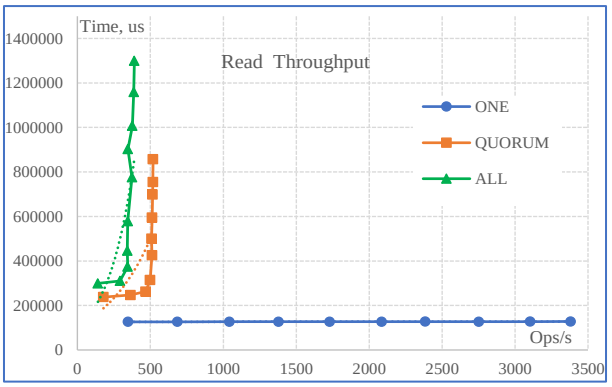


а – час обслуговування централізованого кластеру



б – часова обслуговування розподіленого кластеру

в – пропускна здатність централізованого кластеру



г – пропускна здатність розподіленого кластеру

Рис. 4. Продуктивність операцій читання Cassandra для різних рівнів узгодженності
(**Fig. 4**. Performance of reading operations Cassandra for different levels of consistency)

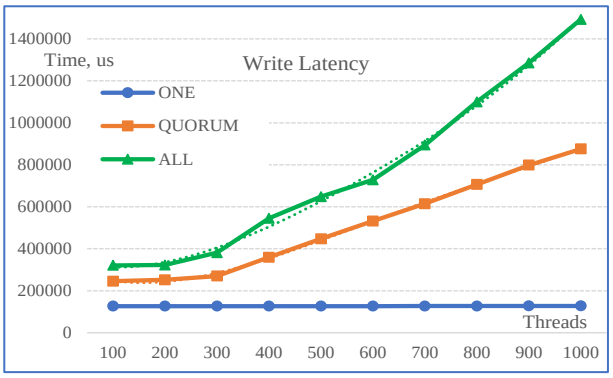
3. Аналіз залежності продуктивності від рівня узгодженості

Результати порівняльного аналізу продуктивності Cassandra при виконання операцій читання та запису зображені на рис. 4, 5.

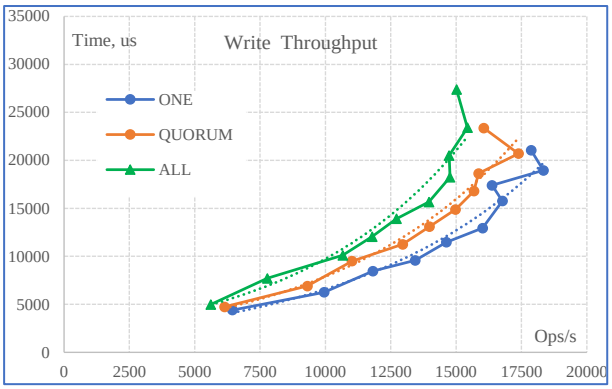
Коли Cassandra налаштована на забезпечення рівня узгодженості ONE, затримка як операцій читання, так і запису менша у середньому на 39% та 21% для централізованого кластеру (рис. 4, а, 5, а) та на 462 та 506% для розподіленого кластеру (рис. 4, б, 5, б) відповідно, ніж середній час відгуку налаштування ALL.

Це свідчить про значний вплив рівня узгодженості на швидкодію бази даних, особливо у випадку глобального розподілення її реплік, а також підтверджує тезу про те, що вибір оптимального рівня узгодженості є одним з найефективніших методів підвищення продуктивності розподілених інформаційних систем.

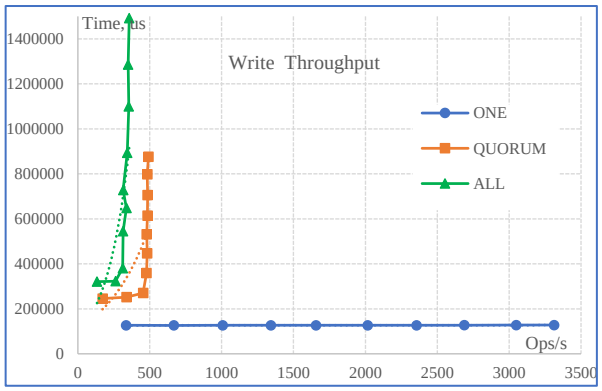
а – часова затримка централізованого кластеру



б – часова затримка розподіленого кластеру



в – пропускна здатність централізованого кластеру



г – пропускна здатність розподіленого кластеру

Рис. 5. Продуктивність операцій запису Cassandra для різних рівнів узгодженності
(**Fig. 5**. Performance of writing operations Cassandra for different levels of consistency)

Налаштування QUORUM демонструє раціональний баланс між затримками та узгодженості даних, однак навіть цей компромісний рівень узгодженості дуже суттєво збільшує час обслуговування запитів читання та запису у випадку глобального розподілення реплік по різних географічних локаціях (на 285% та 300%) у порівнянні з централізованим кластером (19% та 11%). Це обумовлено тим, що для розподіленого кластера мережеві затримки вносять найбільш вагомий вклад (більш ніж 98% замість близько 50% для централізованого кластеру) до загального часу обслуговування запитів читання/запису у порівнянні з часом обробки запитів саме базою даних, що нівелює перевагу швидкого запису. Крім того, важно відмітити той факт, що при розгортанні усіх реплік у межах одного ЦОД операції запису виконуються в середньому на 18% швидше, ніж операції читання, що є відмінною особливістю Cassandra. Навпаки для розподіленого кластера операції запису виконуються в середньому на 2% довше, ніж операції запису, що може бути спричинено традиційно декілька більшою пропускною спроможністю каналу прийому інформації.

Рис. 4, в, г та 5, в, г дозволяють оцінити максимальну пропускну здатність централізованого та розподіленого кластеру. Для централізованого кластура вона сягає 14000 операцій в секунду в найгіршому випадку (читання інформації з найбільш строгим рівнем узгодженості ALL), а для розподіленого кластера фактично зупиняється на рівні 400 операцій в секунду за таких же самих умов. Отримані результати чітко демонструють перевагу у продуктивності що обумовлена використанням найслабшого налаштуваннями узгодженості (ONE) особливо у випадку граничного робочого навантаження.

Результати навантажувального тестування дають певну уяву про продуктивність системи, але не дозволяють розробникам систем точно оцінювати затримку бази даних для всього спектру можливих робочих навантажень. Функція регресії, оцінена за експериментальними даними з використанням методу найменших квадратів може ефективно вирішити цю проблему. У табл. 1 та 2 наведені значення показника R-квадрат, який оцінює точність апроксимації різних функцій регресії.

Таблиця 1 – **Точність регресії експериментальних даних для *централізованого* кластеру Cassandra**

Операція	Рівень узгодженості	Поліноміальна регресія			Лінійна регресія	Експоненціальна регресія
		order=2	order=3	order=4		
Read	ONE	0.9879	0.9898	0.9971	0.9866	0.9597
	QUORUM	0.9732	0.9868	0.9992	0.9697	0.9586
	ALL	0.9642	0.9736	0.9981	0.9626	0.9512
Write	ONE	0.9971	0.9971	0.9977	0.9968	0.9569
	QUORUM	0.9976	0.9996	0.9997	0.9976	0.9407
	ALL	0.9943	0.9996	0.9997	0.9897	0.9542

Таблиця 2 – **Точність регресії експериментальних даних для розподіленого кластеру Cassandra**

Операція	Рівень узгодженості	Поліноміальна регресія			Лінійна регресія	Експоненціальна регресія
		order=2	order=3	order=4		
Read	ONE	0.8652	0.9179	0.9389	0.8301	0.8312
	QUORUM	0.9883	0.9972	0.9976	0.9659	0.9769
	ALL	0.9899	0.9970	0.9975	0.9720	0.9814
Write	ONE	0.9076	0.9076	0.9305	0.8810	0.8813
	QUORUM	0.9915	0.9983	0.9991	0.9754	0.9794
	ALL	0/9963	0.9964	0.9967	0.9577	0.9870

Як слідує з аналізу табл. 1, 2, поліноміальні функції регресії четвертого порядку, наведені нижче (1-12) з найбільш високою точністю описують експериментальні дані.

$$yD_{ALL}^{Read}(x) = 200.82x^4 - 6079.9x^3 + 66136x^2 - 165848x + 411562; \quad (1)$$

$$yD_{QUORUM}^{Read}(x) = 106.71x^4 - 3507x^3 + 40777x^2 - 110191x + 316066; \quad (2)$$

$$yD_{ONE}^{Read}(x) = 1.6149x^4 - 29.622x^3 + 174.2x^2 - 245.82x + 126515; \quad (3)$$

$$yD_{ALL}^{Write}(x) = 172.28x^4 - 4022.4x^3 + 42148x^2 - 76501x + 350165; \quad (4)$$

$$yD_{QUORUM}^{Write}(x) = 148.92x^4 - 4316.7x^3 + 44987x^2 - 111081x + 317546; \quad (5)$$

$$yD_{QUORUM}^{Write}(x) = -1.4094x^4 + 31.096x^3 - 219.89x^2 + 657.15x + 126118; \quad (6)$$

$$yC_{ALL}^{Read}(x) = 27.425x^4 - 564.14x^3 + 3807.9x^2 - 7084.5x + 12314; \quad (7)$$

$$yC_{QUORUM}^{Read}(x) = 16.528x^4 - 323.57x^3 + 2048.9x^2 - 2692.5x + 7922.5; \quad (8)$$

$$yC_{ONE}^{Read}(x) = 11.94x^4 - 248.65x^3 + 1719.6x^2 - 2541.7x + 6483.5; \quad (9)$$

$$yC_{ALL}^{Write}(x) = 0.1047x^4 + 25.676x^3 - 381.66x^2 + 3714.8x + 1623.3; \quad (10)$$

$$yC_{QUORUM}^{Write}(x) = 1.1473x^4 - 10.468x^3 - 60.659x^2 + 2627.1x + 2132.3; \quad (11)$$

$$yC_{QUORUM}^{Write}(x) = -3.0313x^4 + 66.617x^3 - 473.1x^2 + 3016.3x + 1779.1, \quad (12)$$

де yD_{ALL}^{Read}, yD_{QUORUM}^{Read}, yD_{ONE}^{Read}, yD_{ALL}^{Write}, yD_{QUORUM}^{Write}, yD_{ONE}^{Write}– час відгуку Cassandra на читання/запису [мікросекунди] для різних налаштувань узгодженості *розподіленого* кластеру; yC_{ALL}^{Read}, yC_{QUORUM}^{Read}, yC_{ONE}^{Read}, yC_{ALL}^{Write}, yC_{QUORUM}^{Write}, yD_{ONE}^{Write}– час відгуку Cassandra на читання/запису для різних налаштувань узгодженості *централізованого* кластеру; *x* - кількість потоків (наприклад, паралельні запити). Вочевидь, представлені функції регресії (1-12) та їх коефіцієнти є унікальними для нашого експериментального дослідження, а не є універсальними. Інженерам, які проводять моделювання та прогнозування продуктивності розроблювальної системи потрібно буде знайти рівняння, які б відповідали власним результатам навантажувального тестування.

4. Моделювання продуктивності Cassandra при змішаному навантажені та гарантованої узгодженості

Як було зазначено в розділі 2, Cassandra може гарантувати повну узгодженості даних, якщо сума реплік які використовуються для виконання операцій читання та запису перевищує коефіцієнт реплікації. Це означає, що для систем з трьома репліками, що є стандартною архітектурою для багатьох розподілених інформаційних систем, включаючи Facebook, Twitter, тощо, існує 6 можливих налаштувань узгодженості операцій читання/запису, що завжди гарантують повну узгодженість даних:

1) *'Читання ONE – Запис ALL' (1R-3W)*;
2) *'Читання QUORUM–Запис QUORUM' (2R-2W)*;
3) *'Читання ALL – Запис ONE' (3R-1W)*;
4) *'Читання QUORUM – Запис ALL' (2R-3W)*;
5) *'Читання ALL – Запис QUORUM' (3R-2W)*;
6) *'Читання ALL – Запис ALL' (3R-3W)*.

Оскільки всі з наведених варіантів гарантують строгу узгодженість даних, розробники системи можуть бути зацікавлені у виборі такої конфігурації, що забезпечує мінімальний, в середньому, час обслуговування. Результати експериментальних досліджень показують, що чим менше реплік викликається, тим вище швидкодія усього кластеру. Отже, останні три конфігурації є надлишковими та можуть бути виключені з подальшого аналізу.

У свою чергу, затримка відповіді та пропускна здатність Cassandra залежать від поточного навантаження та процентного співвідношення між кількістю операцій читання та запису.

Використовуючи функції регресії (1-12), можливо спрогнозувати середню затримку Cassandra при змішаному навантаженні:

$$y_{1R-3W}(x) = P_{Read} \cdot y_{ONE}^{Read}(x) + P_{Write} \cdot y_{ALL}^{Write}(x); \quad (13)$$

$$y_{2R-2W}(x) = P_{Read} \cdot y_{QUORUM}^{Read}(x) + P_{Write} \cdot y_{QUORUM}^{Write}(x); \quad (14)$$

$$y_{3R-1W}(x) = P_{Read} \cdot y_{ALL}^{Read}(x) + P_{Write} \cdot y_{ONE}^{Write}(x), \quad (15)$$

де P_{Read}, P_{Write} – ймовірності надходження запиту на читання або оновлення (запис) інформації, що залежать від поточної суміші цих операцій $P_{Read} + P_{Read} = 1$.

У табл. 2, 3 наведено вибіркові оцінки затримки обслуговування централізованого та розподіленого кластеру Cassandra для різних конфігурацій, що гарантують строгу узгодженість при змішаному навантаженні. Вони були отримані з використанням (1-15).

Таблиця 2 – **Оцінка часу обслуговування централізованого кластеру Cassandra для різних рівнів узгодженності в залежності від завантаження та пропорції операції читання (R) та запису (W)**

Threads	R/W=10/90%			R/W=30/70%			R/W=50/50%			R/W=90/10%		
	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W
200	7608	7212	**6694**	7357	7478	**7273**	**7107**	7744	7853	**6606**	8276	9012
400	11939	11510	**10368**	11724	11943	**11579**	**8194**	8894	8972	**11081**	13241	15213
600	15759	15149	**14033**	15569	15573	**15488**	**9311**	10083	10222	**15000**	16845	19853
800	20222	18700	**17874**	19684	**18888**	18977	10425	**11256**	11517	**18071**	19449	22285
1000	26973	23768	**22079**	26263	24619	**24366**	11510	**12376**	12790	**24132**	27172	31226

Таблиця 3 – **Оцінка часу обслуговування розподіленого кластеру Cassandra для різних рівнів узгодженності в залежності від завантаження та пропорції операції читання (R) та запису (W)**

Threads	R/W=10/90%			R/W=30/70%			R/W=50/50%			R/W=90/10%		
	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W	1R-3W	2R-3W	3R-1W
200	323543	249747	**144993**	279777	247154	**181569**	236010	244561	**218146**	**148478**	239374	291299
400	466238	351398	**160901**	390813	346606	**228966**	251205	259158	**230485**	**164538**	332231	433162
600	692610	522177	**187956**	566905	517041	**309675**	269555	280783	**248161**	**189789**	501635	674832
800	992658	711575	**218185**	800337	706809	**399767**	290978	308610	**270551**	**223373**	692511	944516
1000	1356378	869083	**243614**	1083391	864258	**475316**	315388	341814	**297031**	**264429**	849784	1170419

У таблицях підкреслено значення мінімальної затримки обслуговування серед трьох можливих варіантів: *1R-3W, 2R-2W* та *3R-1W*.

На перший погляд здається, що в якості компромісного варіанту, що гарантує строгу узгодженість при змішаному навантаженні доречно використовувати конфігурацію 2R-2W, при якої викликається кворум реплік як для виконання читання, так і запису даних.

Однак, як випливає з аналізу табл. 2, 3, конфігурація 2R-2W є оптимальною лише у досить обмеженій області робочого навантаження та процентного співвідношення операцій читання/запису. Так, якщо кількість операцій запису значно превалює над кількістю операцій читання налаштування узгодженості 3R-1W забезпечує найменшу затримку для більшості з можливих робочих навантажень.

Однак із збільшенням відсотка запитів на читання налаштування 1R-3W стає більш оптимальним. Для розподіленого кластера ця межа проходить майже посередині (рис. 6, б).

Для централізованого кластеру границя оптимальності між 3R-1W та 1R-3W знаходиться на межі 30/70 (тобто коли відсоток операцій читання дорівнює 30%, а запису – 70%).

Однак, при збільшенні загальної кількості активних потоків більш ніж 700 в діапазоні пропорції читання/запису від 30/70 до 55/45, оптимальним виявляється і варіант 2R-2W.

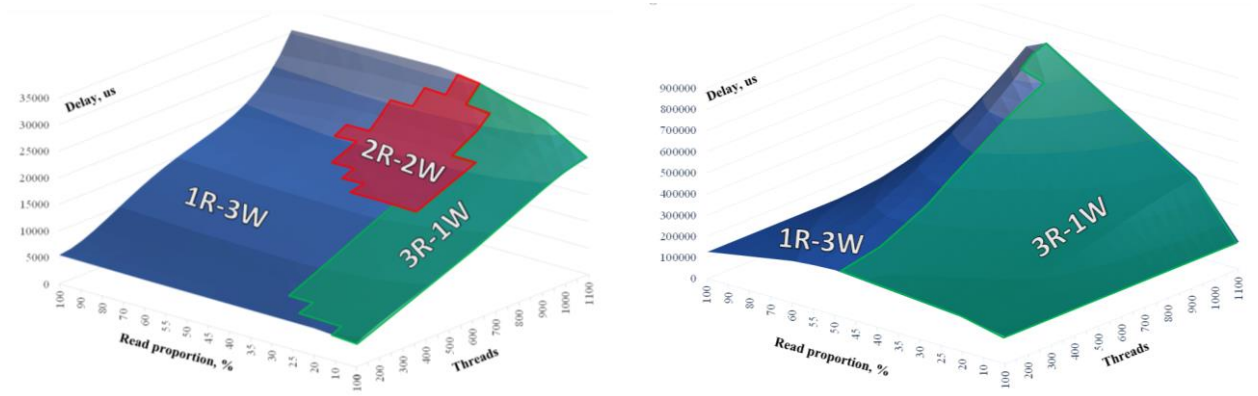
Графіки тривимірної поверхні, представлені на рис. 6, зазначають домени у змішаному робочому навантаженні з оптимальними налаштуваннями узгодженості, що забезпечують найбільшу швидкодію кластера. Оптимальний вибір налаштувань узгодженості для кожного окремого запиту дозволяє зменшити час виконання операцій читання/запису в середньому на 8% для централізованого кластера та на 44% для розподіленого кластера при збереженні гарантії забезпечення строгої узгодженості даних в цілому (рис. 7).

Отримані результати аналітичного моделювання були перевірені за допомогою практичного вимірювання часу обслуговування кластера Cassandra для тих же самих варіантів змішаного робочого навантаження, що наведені на рис. 6 згідно з методикою, що описана у розділі 3.

Результати такої перевірки свідчать про те, що максимальна різниця між теоретичними та експериментальними значеннями часової затримки не перевищують 17% (ця різниця суттєво зменшується при зростанні робочого навантаження), а запропонований підхід дозволяє обрати налаштування узгодженості, що є оптимальними у 92% випадків.

5. Метод оптимального вибору рівня узгодженості даних

Треба зауважити, що отримані у попередньому розділі кількісні результати та обрані функції регресії є унікальними для нашого експериментального дослідження та, вочевидь, не можуть бути оптимальними для всіх можливих варіантів побудови кластеру Cassandra.



а – централізований кластер б – розподілений кластер

Рис. 6. Області навантаження з оптимальними налаштуваннями узгодженості кластеру Cassandra
(Fig. 6. Load areas with optimal Cassandra cluster consistency settings)

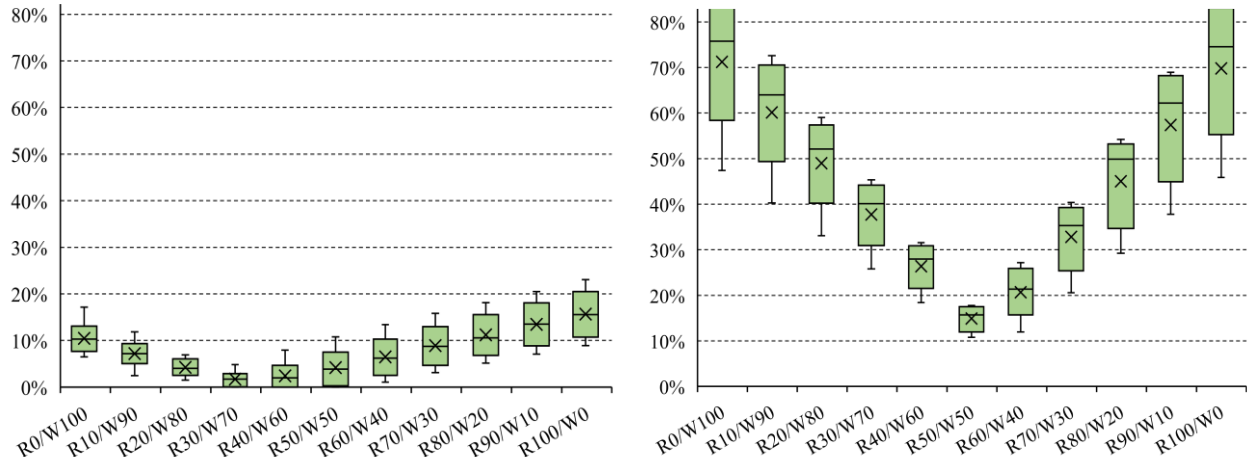


Рис. 7. Зниження часу обслуговування запитів за рахунок оптимального вибору рівня узгодженності в залежності від пропорції операцій читання/запису у порівнянні з конфігурацією 2R-2W
(**Fig. 7.** Reduction of query service time due to the optimal choice of the level of consistency)

Дійсно, продуктивність розподіленої інформаційної системи залежить від багатьох факторів, включаючи розмір і структуру бази даних, використовуване обладнання, кількість реплік та їх географічне розташування, тощо.

У цьому розділі ми узагальнюємо експериментальні дані, які представлені в статті, та пропонуємо метод оптимального динамічного вибору налаштувань узгодженості за рахунок використання результатів навантажувального тестування, проведення якого є обов'язковим етапом розробки розподілених інформаційних систем.

Запропонований метод включає наступні етапи (кроки 1-5 можуть бути виконані одноразово під час навантажувального тестування системи; крок 6 повинен виконуватися безпосередньо під час роботи системи):

1. Розгортання кластеру Cassandra в реальному виробничому середовищі.
2. Модифікація тестових навантажень YCSB для виконання операцій читання та запису, що є специфічними для розробленої інформаційної системи, що є необхідним для оцінки продуктивності системи в реалістичних сценаріях застосування.
3. Виконання навантажувального тестування (бенчмаркингу) кластеру Cassandra при різних робочих навантаженнях (потоків в секунду) з різними налаштуваннями узгодженості згідно сценарію тестування, описаного в розділі 3 та передбачуваним умовам експлуатації.
4. Пошук функцій регресії, які найбільш точно будуть описувати середню затримку виконання операцій читання та запису в залежності від робочого навантаження для різних параметрів узгодженості за результатами практичного вимірювання.
5. Визначення оптимальних налаштувань узгодженості, використовуючи функції (13) - (15) для забезпечення мінімальної затримки Cassandra в залежності від робочого навантаження та співвідношення запитів читання/запису, як описано у розділі 4. В якості результату, розробники системи зможуть визначити домени змішаного робочого навантаження з оптимальними налаштуваннями узгодженості (наприклад, див. рис. 6).
6. Постійний моніторинг поточного навантаження та співвідношення операцій читання/запис під час роботи системи та вибір оптимальних налаштувань узгодженості використовуючи домени робочого навантаження, визначені на попередньому кроці.

Запропонована методологія дозволяє оптимізувати параметри узгодженості динамічно під час роботи кластеру Cassandra для досягнення його

максимальної продуктивності та швидкодії гарантуючи при цьому строгу узгодженість даних.

Висновки

В статті експериментально досліджено взаємодію між різними налаштуваннями узгодженості та продуктивністю розподіленої бази даних Cassandra NoSQL.

Це є важливим елементом досягнення компромісу між доступністю, узгодженістю даних та стійкістю до розділення, що є актуальною проблемою побудови розподілених інформаційних систем та баз даних, особливо за умов глобального розподілення реплік у мережі Інтернет.

Наведені результати показують, що обрані налаштування узгодженості суттєво впливають на час обслуговування та пропускну здатність таких систем на прикладі нереляційної бази даних Cassandra, що необхідно враховуватися під час проектування та експлуатації розподілених інформаційних систем. Дотримання строгої узгодженості даних може значно збільшити час обслуговування клієнтських запитів (приблизно в 1.4 раз для централізованого кластеру та в 6 разів для розподіленого кластеру) та настільки ж погіршити її пропускну здатність.

База даних Cassandra пропонує розробникам унікальну можливість налаштування рівня узгодженості для кожного окремого запиту читання або запису інформації.

Крім того, можна завжди гарантувати строгу узгодженість даних обираючи параметри узгодженості таким чином, щоб загальна сума вузлів, які використовуються під час для читання та запису інформації перевищувала коефіцієнт реплікації. Розробникам розподілених інформаційних систем, де Cassandra використовується в якості сховище NoSQL, рекомендується порівняти ефективність різних параметрів узгодженості при різних робочих навантаженнях та для різних пропорціях між операціями читання та записи.

Це дозволить визначити домени в просторі робочого навантаження, де певні налаштування узгодженості забезпечують максимальну швидкодію.

Одним з головних практичних висновків є той факт, що оптимальні налаштування узгодженості, що максимізують продуктивність Cassandra, суттєво залежать від поточного навантаження та співвідношення запитів на читання та запис. Вони підтверджують ствердження про те, що жодне з налаштувань узгодженості не може завжди гарантувати мінімальну затримку обслуговування.

За результатами проведених досліджень був запропонований метод вибору оптимального налаштування узгодженості для кожній операції читання або запису динамічно під час роботи кластеру Cassandra шляхом моніторингу поточного навантаження та використання результатів бенчмаркінгу, зібраних на етапі навантажувального тестування системи.

References

1. Meier, A. and Kaufmann, M. (2019), *SQL and NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management*, Springer Verlag, Berlin, 229 p.
2. Pritchett, D. (2008), “Base: An Acid Alternative”, *ACM Queue*, Vol. 6, No. 3, pp. 48-55.
3. Brewer, E. (2000), “Towards Robust Distributed Systems”, *Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing*, Portland, USA, pp. 7-8.
4. Gilbert, S. and Lynch, N. (2002), “Brewer’s Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services”, *ACM SIGACT News*, Vol. 33, No. 2, pp. 51-59.
5. Cooper, B., Silberstein, A., Tam, E. and Ramakrishnan, R. (2010), “Sears Benchmarking Cloud Serving Systems with YCSB”, *Proceedings of the 1st ACM Symposium on Cloud Computing*, Indianapolis, USA, pp. 143-154.
7. Abramova, V., Bernardino, J. and Furtado P. (2014), “Testing Cloud Benchmark Scalability with Cassandra”, *Proceedings of the IEEE 10th World Congress on Services*, Anchorage, USA, pp. 434-441.
8. Klein, J., Gorton, I., Ernst, N., Donohoe, P., Pham, K. and Matser C. (2015), “Performance Evaluation of NoSQL Databases: A Case Study”, *Proceedings of the 1st ACM/SPEC Int. Workshop on Performance Analysis of Big Data Systems*, Austin, USA, pp. 5-10.
9. Haughian, G., Osman, R. and Knottenbelt W. (2016), “Benchmarking Replication in Cassandra and MongoDB NoSQL Datastores”, *Proceedings of the 27th Int. Conf. on Database and Expert Systems Applications*, Porto, Portugal, pp. 152-166.
10. Farias, V. A., Sousa, F. R., Maia, J. G. R., Gomes, J. P. P. and Machado J. C. (2018), “Regression based performance modeling and provisioning for NoSQL cloud databases”, *Future Generation Computer Systems*, Vol. 79, pp. 72–81.
11. Karniavoura, F. and Magoutis, K. (2017), “A measurement-based approach to performance prediction in NoSQL systems”, *Proceedings of the 25th IEEE Int. Symposium on the Modeling, Analysis, and Simulation of Computer and Telecom. Systems*, Banff, Canada, pp. 255-262.
12. Cruz, F., Maia, F., Matos, M., Oliveira, R., Paulo, J. and Pereira, J. R. (2017), “Vilaca Resource usage prediction in distributed key-value datastores”, *Proceedings of the IFIP Distributed Applications and Interoperable Systems Conf.* (DAIS'2017), Heraklion, Crete, pp. 144-159.
13. Abdelmoniem, A. M. and Bensaou, B. (2018), “Curbing Timeouts for TCP-Incast in Data Centers via A Cross-Layer Faster Recovery Mechanism”, *Proceedings of the IEEE Conf. on Computer Communications*, Honolulu, HI, pp. 675-683.
14. Libman, L. and Orda, A. (2002), “Optimal retrial and timeout strategies for accessing network resources”, *IEEE/ACM Transactions on Networking*, Vol. 10, No. 4, pp. 551-564.
15. Avizienis, A., Laprie, J.-C., Randell, B. and Landwehr, C. (2004), “Basic concepts and taxonomy of dependable and secure computing”, *IEEE Transactions on Dependable and Secure Computing*, Vol. 1, No. 1, pp. 11-33.
16. Brewer, E. (2012), “CAP twelve years later: How the "rules" have changed”, Computer, Vol. 45, No. 2, pp. 23-29.
17. Gorbenko, A., Romanovsky, A. and Tarasyuk, O. (2019), “Fault tolerant internet computing: Benchmarking and modelling trade-offs between availability, latency and consistency”, *Journal of Network and Computer Applications*, Vol. 146, pp. 1-14.

18. Gorbenko, A. and Romanovsky, A. (2013), “Time-outing Internet Services”, *IEEE Security & Privacy*, Vol. 11, No. 2, pp. 68-71.
19. Chandani, M (2016), *Benchmarking Cassandra and other NoSQL databases with YCSB*, URL: https://github.com/cloudius-systems/osv/wiki/Benchmarking-Cassandra-and-other-NoSQL-databases-with-YCSB
20. Gorbenko, A., Romanovsky, A. and Tarasyuk, O. (2020), “Interplaying Cassandra NoSQL consistency and performance: A benchmarking approach”, *Communications in Computer and Information Science*, Vol. 1279 / Editors: S. Bernardi, et al. Springer Nature. Berlin, pp. 168-184.
21. Gorbenko, A., Kharchenko, V., Tarasyuk, O., Chen, Y. and Romanovsky A. (2008), “The threat of uncertainty in Service-Oriented Architecture”, *Proceedings of the RISE/EFTS Joint International Workshop on Software Engineering for Resilient Systems*, Newcastle, UK, pp. 49-54.

Надійшла (received) 16.06.2021
Прийнята до друку (accepted for publication) 25.08.2021

ВІДОМОСТІ ПРО АВТОРІВ / ABOUT THE AUTHORS

Карпенко Андрій Сергійович – аспірант кафедри комп’ютерних систем, мереж та кібербезпеки Національного аерокосмічного університету ім. М. Є. Жуковського «Харківський авіаційний університет» Харків, Україна;
Andrii Karpenko – PhD student with the Department of Computer Systems, Networks and Cybersecurity, National Aerospace University “Kharkiv Aviation Institute”, Kharkiv, Ukraine,
e-mail: a.karpenko@csn.khai.edu; ORCID ID: http://orcid.org/0000-0003-2789-1168.

Тарасюк Ольга Михайлівна – кандидат технічних наук, доцент, доцент Одеського технологічного університету «ШАГ», Одеса, Україна;
Olga Tarasyuk– Candidate of Technical Sciences, Associate Professor, Associate Professor with the Odessa Technological University STEP, Odessa, Ukraine,
e-mail: O.M.Tarasyuk@gmail.com; ORCID ID: http://orcid.org/0000-0001-5991-8631.

Горбенко Анатолій Вікторович – доктор технічних наук, професор, професор кафедри комп’ютерних систем, мереж та кібербезпеки Національного аерокосмічного університету ім. М. Є. Жуковського «Харківський авіаційний університет» Харків, Україна; Leeds Beckett University, Leeds, Великобританія,
Anatoliy Gorbenko – Doctor of Technical Sciences, Professor, Professor of the Department of Computer Systems, Networks and Cybersecurity, National Aerospace University “Kharkiv Aviation Institute”, Kharkiv, Ukraine; Reader in Computer Science with the School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds, United Kingdom;
e-mail: a.gorbenko@csn.khai.edu; ORCID ID: http://orcid.org/0000-0001-6757-1797.

Исследование согласованности и производительности в нереляционных реплицированных базах данных

А. С. Карпенко, О. М. Тарасюк, А. В. Горбенко

Аннотация. Целью данной статьи является исследование производительности распределенных отказоустойчивых информационных систем и нереляционных хранилищ данных, а также анализ влияния параметров согласованности данных на быстродействие и пропускную способность на примере трех-реплицированного кластера Cassandra. **Результаты.** В статье приведены результаты нагрузочного тестирования (бенчмаркинга) производительности операций чтения и записи кластера Cassandra, реплики которого были развернуты на ресурсах облачного провайдера Amazon Web Services. Представленные количественные результаты показывают, как различные настройки согласованности влияют на производительность Cassandra при различных рабочих нагрузках в условиях, когда все реплики расположены в одном центре обработки данных (ЦОД), или же географически распределены по разным ЦОД. **Вывод.** Предложен метод минимизации временных задержек Cassandra при обеспечении строгой согласованности данных на основе оптимизации настроек согласованности в зависимости от текущей рабочей нагрузки и пропорции между операциями чтения и записи.

Ключові слова: NoSQL; база данных Cassandra; теорема CAP; компромисс; согласованность; задержки; производительность; сравнительный анализ.

Research consistency and perfomance of nosql replicated databases

Andrii Karpenko, Olga Tarasyuk, Anatoliy Gorbenko

Abstract. This paper evaluates performance of distributed fault-tolerant computer systems and replicated NoSQL databases and studies the impact of data consistency on performance and throughput on the example of a three-replicated Cassandra cluster. The paper presents results of heavy-load testing (benchmarking) of Cassandra cluster’s read and write performance which replicas were deployed on Amazon EC2 cloud. The presented quantitative results show how different consistency settings affect the performance of a Cassandra cluster under different workloads considering two deployment scenarios: when all cluster replicas are located in the sane data center, and when they are geographically distributed across different data centers (i.e. Amazon availability zones). We propose a new method of minimizing Cassandra response time while ensuring strong data consistency which is based on optimization of consistency settings depending on the current workload and the proportion between read and write operations.

Keywords: NoSQL; Cassandra database; CAP theorem; compromise; consistency; latency; performance; benchmarking.