УДК 004.896 DOI: 10.31673/2412-4338.2024.036268

Корнага Ярослав Ігорович
Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»
ORCID 0000-0001-9768-2615

Олексій Андрій Васильович
Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»
ORCID 0009-0005-2872-9850

ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ ВЕРТИКАЛЬНОГО ТА ГОРИЗОНТАЛЬНОГО МАШТАБУВАННЯ ОБЧИСЛЮВАЛЬНИХ СИСТЕМ

Анотація. *В статті розглядається актуальна проблема масштабування обчислювальних систем, що є ключовим фактором для забезпечення ефективної роботи сучасних інформаційних систем. Дослідження зосереджено на порівняльному аналізі двох основних підходів до масштабування: вертикального та горизонтального. Метою роботи є визначення оптимального методу масштабування для різних типів навантажень та обчислювальних ресурсів. Для досягнення мети було проведено детальний аналіз існуючих досліджень, а також експериментальні дослідження на основі створених моделей обчислювальних систем. В результаті дослідження було встановлено, що вибір методу масштабування залежить від конкретних вимог до системи, таких як вартість, продуктивність, надійність та гнучкість. Важливим критерієм також є наявність спеціалістів та інфраструктури для побудови того чи іншого рішення, що впливає на можливість вибору в цілому. Було виявлено, що горизонтальне масштабування є більш ефективним для систем з високою динамічністю навантаження, тоді як вертикальне масштабування краще підходить для систем з високими вимогами до обчислювальної потужності окремого вузла. Отримані результати можуть бути використані для розробки рекомендацій щодо вибору оптимального методу масштабування для конкретних застосувань. Результати отримані за допомогою використання існуючих IaaS (Інфраструктура як сервіс) таких як AWS та Azure, а також розгортання тестового серверу в локальних умовах. Перспективи подальших досліджень полягають у розробці гібридних методів масштабування, що поєднують переваги вертикального та горизонтального масштабування, а також у дослідженні впливу нових технологій, таких як контейнеризація та serverless архітектури, на ефективність масштабування.*

Ключові слова: *масштабування, вертикальне масштабування, горизонтальне масштабування, обчислювальні системи, надійність.*

Kornaga Yaroslav
ational Technical University of Ukraine Igor Sikorsky Kyiv Polytechnic Institute
ORCID 0000-0001-9768-2615

Oleksii Andrii
National Technical University of Ukraine Igor Sikorsky Kyiv Polytechnic Institute
ORCID 0009-0005-2872-9850

COMPARATIVE ANALYSIS OF VERTICAL AND HORIZONTAL SCALING METHODS IN COMPUTING SYSTEMS

The article addresses the pressing issue of scaling computing systems, which is a key factor in ensuring the efficient operation of modern information systems. The research focuses on a comparative analysis of two main approaches to scaling: vertical and horizontal. The aim of the study is to determine the optimal scaling

method for different types of workloads and computing resources. To achieve this, a detailed analysis of existing research was conducted, along with experimental studies based on models of computing systems. The study found that the choice of scaling method depends on specific system requirements such as cost, performance, reliability, and flexibility. Another important criterion is the availability of specialists and infrastructure for implementing a particular solution, which affects the overall choice. It was found that horizontal scaling is more effective for systems with highly dynamic workloads, while vertical scaling is better suited for systems with high computational power requirements for individual nodes. The results obtained can be used to develop recommendations for choosing the optimal scaling method for specific applications. The findings were derived using existing IaaS (Infrastructure as a Service) platforms like AWS and Azure, as well as deploying a test server in local conditions. Future research prospects include the development of hybrid scaling methods that combine the advantages of both vertical and horizontal scaling, as well as studying the impact of new technologies such as containerization and serverless architectures on scaling efficiency.

***Keywords**: scaling, vertical scaling, horizontal scaling, computing systems, reliability.*

Постановка проблеми.

Сучасний світ характеризується стрімким розвитком технологій, що породжує безпрецедентний ріст обсягів даних. Цей феномен, відомий як "великі дані", висуває нові вимоги до обчислювальних систем, які повинні не лише ефективно обробляти величезні інформаційні потоки, але й забезпечувати швидкий доступ до них. Одним з ключових аспектів, що визначає здатність системи справлятися з таким навантаженням, є її масштабованість.

Проблема масштабування обчислювальних систем полягає у здатності системи адаптуватися до змінних умов навантаження, зберігаючи при цьому високу продуктивність, доступність та ефективність. Це складне завдання, оскільки воно вимагає балансування протилежних цілей: з одного боку, необхідно забезпечити достатню обчислювальну потужність для виконання всіх необхідних завдань, з іншого - уникнути перевитрати ресурсів та зниження ефективності.

Актуальність цієї проблеми обумовлена кількома факторами. По-перше, безперервне зростання обсягів даних у різних сферах, від науки до бізнесу, створює постійний тиск на обчислювальні ресурси. По-друге, сучасні користувачі очікують миттєвої реакції від інформаційних систем, що висуває високі вимоги до продуктивності. По-третє, різноманітність застосувань обчислювальних систем, від електронної комерції до наукових досліджень, породжує широкий спектр вимог до масштабування. Нарешті, висока вартість обчислювальних ресурсів стимулює пошук оптимальних стратегій масштабування, які дозволяють досягти необхідної продуктивності при мінімальних витратах.

Проблема масштабування тісно пов'язана з вирішенням низки важливих наукових та практичних завдань. Розробка ефективних алгоритмів і структур даних, придатних для обробки великих обсягів інформації на розподілених системах, є одним з ключових напрямків досліджень. Створення гетерогенних обчислювальних систем, що об'єднують різноманітні компоненти для досягнення максимальної продуктивності, також є актуальним завданням. Розробка інструментів для управління розподіленими системами, забезпечення їхньої високої доступності та надійності, а також оптимізація енергоспоживання є невід'ємними аспектами проблеми масштабування.

Аналіз останніх досліджень і публікацій.

Останні дослідження в галузі масштабування обчислювальних систем демонструють зростаючий інтерес до гібридних підходів, які поєднують переваги вертикального та горизонтального масштабування. Це пов'язано з тим, що жоден з цих методів не є універсальним рішенням і оптимальний вибір залежить від конкретних вимог додатку [1]. Дослідники також активно працюють над розробкою нових алгоритмів і інструментів для ефективного управління розподіленими системами, що дозволяє оптимізувати використання ресурсів і підвищити надійність. Особливу увагу приділяється проблемам балансування навантаження, толерантності до відмов та забезпечення безпеки в масштабованих системах

[2]. Крім того, з розвитком хмарних обчислень та мікросервісної архітектури, дослідження все частіше фокусуються на динамічному масштабуванні, яке дозволяє адаптувати ресурси системи до змінних навантажень в реальному часі.

Постановка завдання.

Провести аналіз методів вертикального та горизонтального масштабування обчислювальних систем, порівняти їх ефективність, переваги та недоліки, визначити оптимальні умови застосування кожного методу та розробити рекомендації щодо вибору стратегії масштабування для різних типів навантажень і обчислювальних ресурсів.

Виклад основного матеріалу.

Вертикальне масштабування – це процес збільшення обчислювальної потужності окремого сервера шляхом заміни або додавання більш потужних компонентів. Це відносно простий процес, який не вимагає складних налаштувань мережі. Він дозволяє досягти високої продуктивності для окремих додатків, особливо тих, які вимагають великої кількості пам’яті або обчислювальної потужності одного ядра процесора. Однак, цей метод має свої обмеження. Фізичні характеристики обладнання встановлюють верхню межу для збільшення потужності одного сервера. Крім того, заміна компонентів може бути дорогою, особливо для високопродуктивних систем. Також варто враховувати, що під час заміни компонентів система може бути недоступною. Вертикальне масштабування робить систему залежною від одного сервера: якщо він вийде з ладу, вся система може перестати працювати.

Вертикальне масштабування доцільно застосовувати у випадках, коли додатки потребують значних обсягів оперативної пам'яті або потужності одного ядра процесора – наприклад, у базах даних чи наукових обчисленнях [3]. Також цей метод є ефективним для систем з невеликим навантаженням, де очікується незначне збільшення вимог до ресурсів. Вертикальне масштабування може бути єдиним можливим або економічно вигідним рішенням, якщо горизонтальне масштабування обмежено ліцензійними угодами на програмне забезпечення.

Головні переваги вертикального масштабування полягають у його простоті управління та високій продуктивності для окремих додатків. Крім того, для невеликих систем він може бути економічно вигіднішим. Проте, цей метод має і свої недоліки. По-перше, існує фізична межа збільшення потужності одного сервера. По-друге, заміна компонентів може бути дорогою, особливо для високопродуктивного обладнання. Під час заміни компонентів система може бути недоступною, що призводить до простоїв. І, нарешті, вертикальне масштабування робить систему залежною від одного сервера: якщо він вийде з ладу, вся система може перестати працювати.

Горизонтальне масштабування є підходом до збільшення обчислювальної потужності системи шляхом додавання додаткових незалежних одиниць обладнання, таких як сервери або віртуальні машини. На відміну від вертикального масштабування, де потужність системи збільшується за рахунок удосконалення одного пристрою (наприклад, додаванням оперативної пам'яті чи процесорів), горизонтальне масштабування дозволяє масштабувати систему шляхом розширення її інфраструктури.

Основна концепція горизонтального масштабування полягає в тому, що замість того, щоб покладатися на один потужний сервер, система розподіляє свої завдання між кількома менш потужними серверами. Це дає змогу розподілити навантаження, забезпечуючи більш рівномірне використання ресурсів та підвищуючи загальну продуктивність.

Як це працює на практиці? При горизонтальному масштабуванні система включає в себе кілька серверів, які працюють разом як єдине ціле. Завдання та запити від користувачів розподіляються між цими серверами. Для цього часто використовуються спеціальні інструменти або алгоритми, які визначають, на який сервер направити конкретний запит.

Такий підхід дозволяє уникнути перевантаження окремого сервера і знижує ризик збоїв у роботі системи.

Горизонтальне масштабування особливо ефективне в середовищах, де очікується велика кількість одночасних запитів або де обробляються великі обсяги даних. Наприклад, у випадку з веб-сервісами, які обслуговують мільйони користувачів, горизонтальне масштабування дозволяє забезпечити швидкий доступ до ресурсу без затримок, незалежно від кількості відвідувачів.

Ключовою перевагою горизонтального масштабування є його гнучкість. Систему можна поступово розширювати, додаючи нові сервери у міру зростання потреб. Це робить горизонтальне масштабування привабливим варіантом для бізнесів, які прагнуть забезпечити стабільну та надійну роботу своїх інформаційних систем навіть при високих навантаженнях.

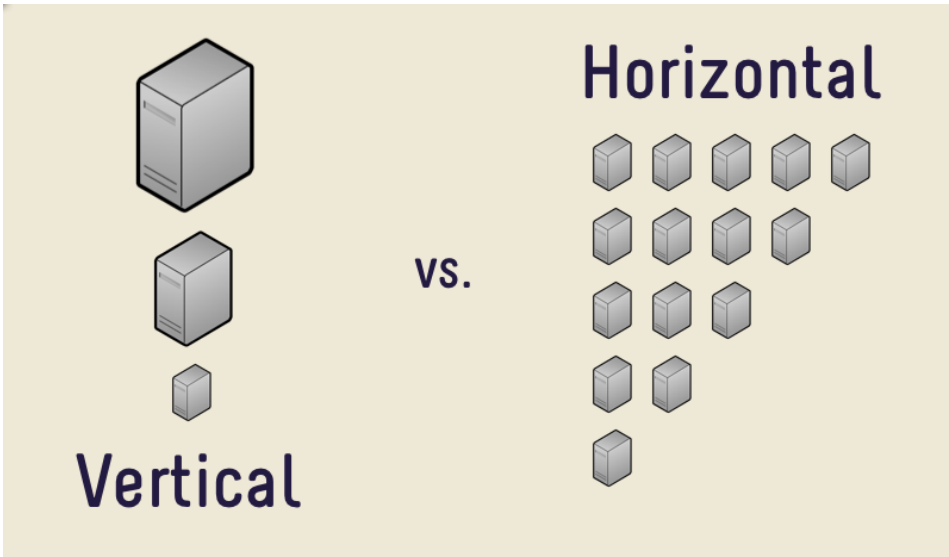

Рис. 1. Схематичне зображення типів масштабування

Для об’єктивної порівняльної оцінки вищеописаних методів масштабування було вирішено вивести узагальнені критерії, згідно яких було проведене дослідження. Дослідження полягало як в опрацюванні існуючих робіт, так і в проведенні експериментальних вимірювань.

Ціна та складність імплементації є одним з ключових факторів при виборі стратегії масштабування. Проблема адаптації системи до збільшених навантажень в своєму корені є проблемою правильного перерозподілу наявних ресурсів. Команда інженерів, які займаються підтримкою та впровадженням рішень є таким ж ресурсом, як фінансові кошти та серверне обладнання, тож вплив складності задачі, а значить витрат часу потрібно враховувати в тому ж пріоритеті. В даному контексті вертикальне масштабування виграє в своїй простоті, тому що для усереднених задач, які ми розглядаємо в контексті цієї роботи воно полягає тільки в зміні об’єму доступних операційних ресурсів системи. Якщо взяти за приклад розгортання системи в IaaS рішеннях (як наприклад AWS) то вертикальне масштабування стає задачею, яка не потребує окремої підготовки рішення. В випадку ж горизонтального масштабування додатковою складністю виступає потреба підготування додатку до можливого паралельного виконання запитів. Для цього потрібно імплементовувати програмні рішення, які унеможливлять проблеми на кшталт Race Conditions [7]. Також горизонтально розширена система складніша в тестуванні та потребує трасування. В контексті оцінки ціни фізичних ресурсів варто враховувати, що це суто специфічна оцінка до конкретних проектів та систем, тож не є можливим чітко дати порівняльну оцінку загальним методам в цьому контексті. Ціна імплементації може варіюватися від факту наявності власних фізичних серверів, договорів з провайдерами IaaS, регіонів розгортування, тощо. Тим не менш важливо враховувати цей критерій в конкретних випадках оцінки та проектування масштабованих архітектур.

Границі масштабування є критичним фактором вибору підходу до розширення системи. Суттєвою проблемою вертикального масштабування є верхня межа обчислювальної потужності одного конкретного сервера[10]. Ця межа може залежати як від фізичних лімітів доступного обладнання, так і особливостей самої системи. В той час горизонтальне масштабування не має цих обмежень та, якщо не брати до уваги окремі випадки, не обмежене в розширенні. За умови вірно підібраної інфраструктури та реалізації горизонтальна структура може розширюватися без логічних обмежень.

Гнучкість є критерієм, який позначає здатність існуючої системи до динамічних змін. В випадку з вертикальним масштабуванням збільшення доступних ресурсів може потягнути за собою потребу в фізичному переміщенні контексту виконання, що тягне за собою потребу в певному часі переходу. Для систем, які не можуть собі дозволити бути не активними на певному проміжку часу це виступає важливим критерієм. Горизонтальне масштабування дає можливість запускати нові репліки сервера[9] не вимикаючи існуючі сервери. Це дає можливість збільшувати ресурси системи без потреби вводу в недоступний стан.

Як можна побачити вище, **доступність** є важливим критерієм при плануванні розробки програмної системи[8]. Це важливий показник її надійності та ефективності, який характеризує здатність системи виконувати свої функції без перерв та збоїв протягом певного періоду часу. Іншими словами, це міра того, наскільки часто користувачі можуть отримати доступ до системи та використовувати її послуги. Важливим критерієм доступності є поведінка системи під час навантаження, що напряму пов’язано з питанням масштабування. Нижче приведені та деталізовані результати експериментального дослідження щодо поведінки та метрик системи. Дослідження проведено на базі просто веб-застосунку реалізованого на технології .net core, базою даних Postgres, розгорнутого через контейнери Docker. Тест проведений за допомогою JMeter.

```
version: '3.8'

services:
  webapp:
    image: mcr.microsoft.com/dotnet/aspnet:7.0
    container_name: webapp
    ports:
      - "5000:80"
    environment:
      - ASPNETCORE_ENVIRONMENT=Production
      - ConnectionStrings__DefaultConnection=Server=postgres;Port=5432;Database=mydb;User Id=postgres;Password=mysecretpassword;
    depends_on:
      - postgres
    volumes:
      - ./app:/app
    working_dir: /app
    command: ["dotnet", "MyApp.dll"]

  postgres:
    image: postgres:15-alpine
    container_name: postgres
    environment:
      POSTGRES_DB: mydb
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: mysecretpassword
    ports:
      - "5432:5432"
    volumes:
      - pgdata:/var/lib/postgresql/data

volumes:
  pgdata:
```

Рис. 2. Docker-compose.yml використаний для тесту

Для порівняння взяті наступні метрики:

- Продуктивність системи (запити на секунду, RPS)
- Час відгуку (мс), вимірюється середній час відгуку системи на запит від клієнта при збільшенні навантаження.
- Стійкість до відмов (відсоток відмов), вимірюється частка відмов у обробці запитів при виникненні проблем з обладнанням.
- Ефективність використання ресурсів (відсоток), порівнюється середній відсоток використання ЦП, оперативної пам'яті та інших ресурсів при різних рівнях навантаження.

Таблиця 1

Результати навантаженого тесту

Метрика	Вертикальне масштабування	Горизонтальне масштабування
Продуктивність системи (RPS)	15000 RPS	50000 RPS
Час відгуку (мс)	150 мс (10000 RPS)	100 мс (10000 RPS)
	250 мс (15000 RPS)	120 мс (50000 RPS)
Стійкість до відмов (відсоток)	5% відмов	1% відмов
Ефективність використання ресурсів (відсоток)	85% використання ресурсів	70% використання ресурсів

Як можна побачити з результатів, горизонтально масштабована система показує кращі результати при умовах високого навантаження.

Таблиця 2

Результати ненавантаженого тесту

Метрика	Вертикальне масштабування	Горизонтальне масштабування
Час відгуку (мс)	40 мс (100 RPS)	43 мс (100 RPS)
Стійкість до відмов (відсоток)	0% відмов	0% відмов
Ефективність використання ресурсів (відсоток)	2% використання ресурсів	3.4% використання ресурсів

Тест при умовах низького навантаження показує, що при прогнозованому невисокому навантаженні вертикально масштабована система більш ефективна за рахунок меншого об’єму ресурсів, які потребують оркестрації та розробки.

Висновки.

Обираючи між вертикальним і горизонтальним масштабуванням, необхідно враховувати специфіку кожного з методів та конкретні вимоги системи. Вертикальне масштабування, хоча і є простішим у реалізації, обмежене фізичними можливостями обладнання. Горизонтальне масштабування, своєю чергою, пропонує більшу гнучкість та масштабованість, але вимагає складнішого програмного забезпечення та інфраструктури. Таким чином, вибір між вертикальним та горизонтальним масштабуванням залежить від специфіки завдань та вимог до системи. Вертикальне масштабування підходить для систем з обмеженими потребами в ресурсах, тоді як горизонтальне масштабування забезпечує стабільну роботу при зростаючому навантаженні та великій кількості одночасних запитів. Ретельний аналіз потреб системи і прогнозованого розвитку допомагає прийняти обґрунтоване рішення щодо вибору відповідного методу масштабування.

Проведене дослідження підтвердило висунені гіпотези про доцільність використання горизонтального масштабування в умовах високого навантаження.

Також варто додати, що сучасні тенденції свідчать про зростаючу популярність гібридних підходів, які поєднують переваги обох методів. Це дозволяє створювати системи, які здатні адаптуватися до змінних навантажень та забезпечувати високу доступність.

Список використаних джерел

1. A. Kwan, J. Wong, H.A. Jacobsen and V. Muthusamy, "HyScale: Hybrid and Network Scaling of Dockerized Microservices in Cloud Data Centres," 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 2019, pp. 80-90
2. Afzal, S., Kavitha, G. Load balancing in cloud computing – A hierarchical taxonomical classification. J Cloud Comp 8, 22 (2019).
3. S. Spinner et al., "Runtime Vertical Scaling of Virtualized Applications via Online Model Estimation," 2014 IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems, London, UK, 2014, pp. 157-166
4. What are race conditions?: Some issues and formalizations 1992. URL: https://dl.acm.org/doi/abs/10.1145/130616.130623
5. Bhagwan, R., Savage, S., Voelker, G.M. (2003). Understanding Availability. In: Kaashoek, M.F., Stoica, I. (eds) Peer-to-Peer Systems II. IPTPS 2003. Lecture Notes in Computer Science, vol 2735. Springer, Berlin, Heidelberg.
6. Sivasubramanian S. Szymaniak M. Pierre G. van Steen M. (2004). Replication for web hosting systems.ACM Computing Surveys, 36(3), 291–334.
7. Sakurai, H., Phung-Duc, T. Scaling limits for single server retrial queues with two-way communication. Ann Oper Res 247, 229–256 (2016).