УДК 004.424

**М. А. Перебора**
**Н. В. Добровольська**

# МЕТОДИ ОПТИМІЗАЦІЇ НЕРІВНОМІРНОГО НАВАНТАЖЕННЯ І РОЗПОДІЛУ РЕСУРСІВ У ВІРТУАЛЬНИХ СЕРЕДОВИЩАХ

Вінницький національний технічний університет

***Анотація***

*У цій роботі досліджено теоретичні основи та практичні методи оптимізації нерівномірного навантаження і розподілу ресурсів у віртуалізованих середовищах. Проаналізовано сучасні підходи, зокрема балансування навантаження, автоматичне масштабування та оркестрацію контейнерів, які забезпечують ефективне функціонування інформаційних систем за умов змінної інтенсивності запитів. Наведено приклади реалізації методів та їхній вплив на продуктивність, стабільність і економічність IT-інфраструктури.*

**Ключові слова:** *віртуалізація, розподіл ресурсів, балансування навантаження, масштабування, оркестрація, оптимізація.*

***Abstract***

*This work examines the theoretical foundations and practical methods of optimizing uneven load and resource distribution in virtualized environments. Modern approaches, including load balancing, automatic scaling, and container orchestration, are analyzed to ensure the efficient functioning of information systems under conditions of variable request intensity. Examples of implementation of methods and their impact on the performance, stability, and cost-effectiveness of IT infrastructure are presented.*

**Keywords**: *virtualization, resource allocation, load balancing, scaling, orchestration, optimization.*

## Вступ

У сучасних інформаційних системах, побудованих на основі віртуалізації та контейнеризації, виникає потреба в динамічному управлінні ресурсами. Через змінне та часто нерівномірне навантаження критично важливо забезпечити ефективний розподіл обчислювальних потужностей між віртуальними машинами, контейнерами та фізичними вузлами. Неоптимальне навантаження призводить до зниження продуктивності, надмірних витрат та порушення стабільності сервісів. Для уникнення таких ризиків застосовують низку методів оптимізації, що ґрунтуються на адаптивному управлінні навантаженням і динамічному масштабуванні ресурсів.

## Основна частина

Одним із ключових підходів до оптимізації ресурсів у віртуалізованих середовищах є балансування навантаження, що забезпечує рівномірний розподіл вхідних запитів або обчислювальних задач між кількома обчислювальними одиницями — віртуальними машинами, контейнерами або фізичними серверами. Основна мета цього методу полягає у мінімізації часу відповіді та уникненні перевантаження окремих вузлів системи. Балансування може реалізовуватись за допомогою різних алгоритмів:

- Round Robin — рівномірний розподіл запитів по колу;
- Least Connections — направлення запитів до вузлів з найменшою кількістю активних з’єднань;
- Weighted Distribution — з урахуванням продуктивності або доступності кожного вузла.

Використання алгоритмів Least Connections та Weighted Distribution, згідно з даними досліджень [1] (Arxiv, ITConvergence, VShosting), дозволяє зменшити середній час обробки запитів до 15–25 % та підвищити пропускну здатність систем. Популярними інструментами реалізації балансування є HAProxy, NGINX, AWS Elastic Load Balancer та Azure Load Balancer, які також забезпечують автоматичне перенаправлення трафіку у разі збою певного вузла, що підвищує загальну стійкість сервісів [2].

Другим важливим методом є автоматичне масштабування, що полягає у динамічному коригуванні кількості обчислювальних одиниць або доступних їм ресурсів відповідно до навантаження. Масштабування реалізується у двох формах:

- горизонтальне масштабування — додавання або вилучення екземплярів контейнерів або віртуальних машин;
- вертикальне масштабування — збільшення або зменшення ресурсів (CPU, RAM) для кожного екземпляра.

За даними аналітичних досліджень, застосування автошкалування у хмарних середовищах дозволяє скоротити витрати до 40 %, а у випадку комбінованого використання стратегій Scale-Up/Scale-Down — до 63 % на обчислювальні ресурси і до 69 % на зберігання [3] (Cloudkeeper, AWS). Такі сервіси, як AWS Auto Scaling, Google Compute Engine Autoscaler [4] та Kubernetes Horizontal Pod Autoscaler (HPA), забезпечують адаптивність системи до пікових навантажень, одночасно мінімізуючи простої.

Окрім масштабування, сучасні системи управління контейнерами, зокрема Kubernetes, надають потужні інструменти оркестрації, що дозволяють контролювати споживання ресурсів на рівні окремих контейнерів. Механізми resource requests та limits визначають межі використання ресурсів, тоді як класи якості обслуговування (QoS) дозволяють пріоритезувати критичні сервіси. Вбудований планувальник Kubernetes приймає рішення про розміщення контейнерів з урахуванням навантаження на хости, що дозволяє уникнути ефекту «шумного сусіда» та забезпечити ефективне використання ресурсів у кластері [5]. Дослідження показують, що контроль ресурсів на рівні контейнерів підвищує продуктивність кластерів на 20–35 %.

У контексті статистичного аналізу, симуляції з використанням CloudAnalyst і CloudSim підтверджують ефективність адаптивних алгоритмів розподілу [6]. Наприклад, застосування динамічного та зваженого балансування демонструє переваги над статичними стратегіями за рахунок скорочення затримок. Новітні підходи, засновані на federated learning (зокрема, моделі DA-DBL та RO-COA), забезпечують підвищення ефективності використання ресурсів та зниження latency у розподілених хмарних середовищах [7].

Застосування комплексного підходу, що об'єднує балансування навантаження, автоматичне масштабування та оркестрацію, дозволяє створити самоналагоджувальні інфраструктури, здатні адаптуватися до динаміки робочого навантаження. Доповнення таких архітектур інструментами моніторингу, як-от Prometheus і Grafana, забезпечує прозорість у роботі кластерів та дозволяє своєчасно реагувати на зміни [8]. За результатами досліджень, інтегроване використання трьох методів може підвищити ефективність роботи IT-інфраструктури до 50 %.

## Висновки

Оптимізація нерівномірного навантаження та розподілу ресурсів у хмарних і віртуалізованих середовищах є ключовою умовою стабільної роботи систем і раціонального використання інфраструктури. Основними методами є балансування навантаження, автоматичне масштабування та оркестрація контейнерів. Їх поєднання дозволяє створювати адаптивні та економічно ефективні рішення. Алгоритми Round Robin, Least Connections та Weighted Distribution забезпечують гнучкий розподіл запитів, тоді як масштабування дозволяє реагувати на зміну навантаження, знижуючи витрати. Оркестрація контейнерів (зокрема, у Kubernetes) забезпечує стабільну роботу критичних сервісів за рахунок точного керування ресурсами.Аналіз статистичних даних підтверджує ефективність цих підходів. Перспективи подальшої оптимізації пов'язані з використанням прогнозних моделей, штучного інтелекту та машинного навчання для підвищення точності розподілу навантаження в реальному часі.

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ


1. Comparative Study of Load Balancers in Production // vshosting.eu [Електронний ресурс] – Режим доступу: https://www.vshosting.eu
2. Auto Scaling and Elastic Load Balancing // AWS Documentation [Електронний ресурс] – Режим доступу: https://docs.aws.amazon.com
3. AWS Cost Optimization Report // Cloudkeeper [Електронний ресурс] – Режим доступу: https://www.cloudkeeper.com
4. Google Compute Engine Autoscaler Overview // Google Cloud [Електронний ресурс] – Режим доступу: https://cloud.google.com/compute/docs/autoscaler
5. Resource Management for Pods and Containers // Kubernetes Documentation [Електронний ресурс] – Режим доступу: https://kubernetes.io/docs
6. Calheiros R. N., Ranjan R., Beloglazov A., De Rose C. A. F., Buyya R. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments // Software: Practice and Experience. – 2011. – Vol. 41(1). – P. 23–50.
7. Federated Learning for Cloud Resource Allocation // Nature.com [Електронний ресурс] – Режим доступу: https://www.nature.com/articles/cloud-ml-allocation
8. Monitoring Distributed Systems with Prometheus and Grafana [Електронний ресурс] – Режим доступу: https://prometheus.io ; https://grafana.com



***Перебора Микола Анатолійович*** — студент групи 1КІ-24м, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький Національний Технічний Університет, Вінниця.

***Добровольська Наталія Вікторівна*** — доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, м. Вінниця, email: dobr_n_v@vntu.edu.ua

***Perebora Mykola Anatoliovich*** — student of group 1KI-22ms, faculty of information technologies and computer engineering, Vinnytsia National Technical University, Vinnytsia.

***Dobrovolskaya Natalia Viktorivna*** — Associate Professor at the Department of Computer Engineering, Vinnytsia National Technical University, Vinnytsia, Ukraine, email: dobr_n_v@vntu.edu.ua