DOI: https://doi.org/10.15276/hait.08.2025.31
UDC 004.89:004.272.4:681.518

Game-theoretic method of decentralized load balancing in microservice architectures

Andrii I. Hryshchenko[1)]
ORCID: https://orcid.org/0009-0007-1191-7948; andrew.hryshchenko@gmail.com
Nataliia O. Komleva[2)]
ORCID: ,https://orcid.org/0000-0001-9627-8530; komleva@op.edu.ua. Scopus Author ID: 57191858904
[1)] Lowe's Companies, Inc., Mooresville, NC, USA
[2)] Odesa Polytechnic National University, 1, Shevchenko Ave, Odesa, 65044, Ukraine

ABSTRACT

The paper presents a game-theoretic method of decentralized load balancing in microservice architectures, aimed at improving the efficiency of request distribution among service instances without using centralized controllers. The main idea is to represent the balancing process as a non-cooperative potential game in which each microservice is considered an autonomous agent seeking to minimize its own cost function. Unlike traditional algorithms such as Round Robin and Least Connections, the proposed approach is based on adaptive adjustment of agent strategies depending on the current state of the system, which ensures the achievement of Nash equilibrium and a stable load distribution.

The mathematical model of game-theoretic method of decentralized load balancing takes into account the intensity of the request flow, the throughput of each node, and the quadratic component of the cost associated with resource overload. To optimize the decision-making process, a stochastic Softmax-update dynamics are used, which approximate the gradient descent of the potential function. This allows the system to gradually balance the load even in the presence of asynchronous updates and communication delays between nodes. It has been proven that the process converges to a stationary state in polynomial time, ensuring scalability and predictable behavior in large distributed environments.

An experimental study conducted in the SimPy simulation environment demonstrated that the proposed method significantly outperforms classical algorithms in key metrics. Under peak load, the Game-Theoretic Load Balancer algorithm reduced the average system response time compared to the Round Robin and Least Connections algorithms. The standard deviation of processor utilization decreased, indicating a more balanced workload distribution and the absence of overloaded computing nodes.

The obtained results confirm the analytical stability, convergence, and practical effectiveness of the game-theoretic approach. The developed method ensures adaptive self-regulation of the system, minimizes the risk of overload, and increases the reliability of microservice architectures. Future research should focus on integrating game-theoretic method of decentralized load balancing with cloud orchestrators and extending the model to multi-level games in hybrid computing environments.

Keywords: Load balancing; microservice architecture; game theory; Nash equilibrium; distributed systems; software engineering; resource optimization

For citation: **Hryshchenko A. I., Komleva N. O. "Game-theoretic method of decentralized load balancing in microservice architectures". *Herald of Advanced Information Technology*. 2025; Vol.8 No.4: 488–496. DOI: https://doi.org/10.15276/hait.08.2025.31**

INTRODUCTION

Modern software systems increasingly adopt microservice architectures (MSA) due to their scalability, modularity, and deployment flexibility. In these systems, application logic is decomposed into a network of loosely coupled services that interact over lightweight protocols. This decomposition enhances system resilience and development agility but introduces substantial challenges in runtime orchestration, particularly in the domain of load balancing.

As user demand fluctuates and service instances dynamically scale, efficient request distribution becomes critical to ensure low latency, high availability, and resource efficiency. However, classical load balancing strategies, such as Round Robin (RR) or Least Connections (LC) often operate without sufficient context and fail to adapt to can heterogeneous server capacities or temporal load spikes. These shortcomings motivate the search for more responsive and theoretically grounded approaches.

In cloud-native environments, where elasticity and dynamic scaling are key, the lack of coordination between service replicas often leads to inefficiencies or bottlenecks. Additionally, the decentralized nature of microservices complicates the task of centralized control, making it difficult to capture system-wide state in real time. As a result, balancing algorithms must operate under partial observability, be robust to delays and asynchronous updates, and adapt to rapidly changing conditions.

Recent advances in distributed systems suggest that modeling service instances as rational agents engaged in strategic decision-making may offer new insights. In this paradigm, load balancing is not

merely a control task but a distributed optimization problem where each agent seeks to improve its own performance under limited information. Game theory provides a powerful mathematical framework to analyze such interactions and offers convergence guarantees under various equilibrium dynamics.

This perspective opens the door to novel approaches that combine theoretical rigor with practical applicability in dynamic service-oriented environments.

RELATED WORKS AND PROBLEM STATEMENT

The simplest approaches to request distribution in distributed systems are deterministic or heuristic algorithms such as RR, Weighted RR, LC, and others. These methods provide basic distribution (for example, RR sends requests to each server in turn equally) but do not take into account current changes in the system state [1]. As a result, in environments with uneven load, they may perform poorly: static schemes such as RR do not respond to traffic fluctuations, while more dynamic algorithms (such as LC or Least Response Time) remain merely reactive and can cause short-term oscillations and instability under sudden load spikes. In particular, during traffic surges, LC may redirect excessive requests to a less loaded node with a delay, which leads to an oscillation effect – alternating overload of different servers without achieving a stable balance [2]. In modern microservice architectures with unpredictable load profiles and strict latency requirements, such simple heuristics are considered insufficient.

A systematic approach to selecting architectural patterns for IoT and distributed systems was proposed to formalize reliability, scalability, and adaptability criteria using a weighted evaluation model, which can be effectively applied to microservice environments [3].

The adequacy of feedback and state estimation in such distributed systems also depends on the informational integrity of collected metrics, as demonstrated in [4], where Shannon entropy–based evaluation was used to detect data loss and ensure the stability of decision systems under uncertain monitoring conditions.

To overcome the limitations of traditional algorithms, researchers have turned to the mathematical framework of game theory, which allows modeling the interaction of independent agents in a distributed system [5]. The application of game-theoretic models to resource management tasks originated in the field of computer networks: as early as the 1990s, traffic routing was formalized as a game in which network nodes act either non-cooperatively, each optimizing its own metrics, or cooperatively, negotiating a joint solution [6], [7]. It was shown that uncoordinated, selfish behavior leads to the loss of global efficiency, which is quantitatively characterized by the metric «price of anarchy» – the ratio of performance in equilibrium to optimal values. In particular, studies [8], [9] demonstrated that the Nash equilibrium (NE) in network games can significantly differ from the centralized optimum. Nash equilibrium, first formalized in the classical work [10], is a system state in which no player can improve their payoff (for example, reduce response time) by unilaterally changing their strategy. In the context of load balancing, NE corresponds to a self-regulated distribution of requests among servers where each service has chosen a strategy beneficial to itself. However, such a non-cooperative equilibrium may not be socially optimal – that is, the response time or other metrics at equilibrium may be worse than under centralized planning [11].

Similar principles of agent interaction and shared-resource optimization have been observed in software engineering models [12], where client–resource class coordination is implemented through queue-based mechanisms ensuring consistent access and minimizing contention. These concepts align with the equilibrium perspective in decentralized load balancing. Practical studies also confirm that the efficiency of processing queues under maximum server load can be significantly improved through adaptive scheduling and priority control, which ensures stable throughput in distributed environments [13].

In distributed computing systems, game-theoretic models of load balancing began to appear in the early 2000s. For example, in [14], static task distribution in a heterogeneous cluster system was formulated as a non-cooperative game: each node (user) acts independently, seeking to minimize its average processing time. The result is a «user-optimal» equilibrium in which no user can gain by unilaterally changing their decision – effectively, a balanced state is achieved, though without guarantees of global optimality. Thus, early studies established two distinct approaches: the non-cooperative (selfish) one – for scenarios where nodes belong to different users interested primarily in their own benefit, and the cooperative one – for cases where coordinated planning is possible to achieve a global optimum. Later works combined both paradigms: in [15], non-cooperative interaction between service instances was used for load

balancing in microservices, but for coordinating routing across service chains, a multi-party agreement based on the Nash bargaining solution (a cooperative element) was introduced. Another recent approach was proposed in [16], which considered two efficiency metrics simultaneously – performance and processing cost. In this model, each player (node) minimizes its own utility function combining response time and resource cost, achieving a «cost-aware» load distribution that balances performance and resource price.

A separate line of research is devoted to hierarchical and incentive-based game models. Stackelberg games (leader–follower games) make it possible to model situations where one participant acts as a leader and others react to its strategy. This approach captures multi-level system control and can improve efficiency if the leader chooses a strategy considering followers' responses. A notable example is [17], where a Stackelberg game is implemented between a request dispatcher (leader) and a set of servers (followers) in a cloud data center. The balancer first evaluates the satisfaction coefficient of each server and selects the optimal one for a new task, after which servers process assigned requests. This two-stage strategy (SGMLB algorithm) increases average resource utilization to about 60% and reduces failed (rejected) tasks by 47% compared to random distribution. Another hierarchical approach is dynamic pricing: load balancing is formulated as a game between the cloud provider and consumers, where the provider sets the price for resources, and users choose which server to send their requests to based on the price. In model [18], each host assigns a service rate (cost) for processing requests, and each user aims to minimize their expenses by choosing the most advantageous offer – resulting in an equilibrium that aligns both parties' interests. Such pricing mechanisms effectively implement auction or market-based regulation and can be used for dynamic load balancing in cloud platforms. More complex multi-level games have also been explored: for example, [19] proposed a three-level game for joint resource management in cloud and fog nodes. At the top level, this hierarchical model employs Stackelberg interaction between the cloud and the edge, while lower levels include subgames for solving local optimization tasks, achieving coordination across system layers.

Since participants in real systems make decisions asynchronously (lacking a shared synchronization mechanism), ensuring convergence of the game-theoretic algorithm under asynchronous strategy updates becomes a critical requirement. The theory of potential games states that if the load balancing problem can be formulated as an exact potential game, then any sequence of improving actions by players (even if they update strategies sequentially or in arbitrary order) will lead the system to Nash equilibrium. In other words, there exists a global potential function that monotonically decreases with selfish player actions, ensuring convergence to a stationary distribution (not necessarily globally optimal but stable) [20]. For practical implementation, stochastic best-response or reinforcement-learning approaches are often used. For instance, [21] proposed an adaptive algorithm where agents (servers) gradually learn to balance load through trial and error without centralized control. Each agent observes only local information about its state and randomly redirects part of the tasks to other nodes; over time, such multi-agent learning algorithms converge to more efficient distributions, even if updates occur asynchronously.

An additional example of hybrid systems operating under uncertainty can be found in [22], where ensemble neural classifiers were combined with statistical methods for real-time medical diagnostics, achieving stability and high accuracy despite small and noisy datasets – a principle conceptually aligned with decentralized adaptive optimization. Recent studies also demonstrate that AI-based decision-support modules integrated into distributed architectures can enhance scalability and improve the adaptability of microservice systems [23].

The results of [24] showed that, with proper parameter selection, a simple local learning rule can ensure high utilization of all servers while avoiding overload of individual nodes.

The literature review shows that despite the variety of approaches, the problem of dynamic load balancing in microservice systems remains far from fully solved. Classical heuristics (RR, LC, etc.) are too simple and do not guarantee either optimal distribution or stability under varying loads [25]. Another challenge lies in the practical implementation of proposed methods. Some algorithms that exhibit good convergence in theory may prove sensitive to parameters in real deployments. For example, the distributed gradient-based load balancing algorithm [26] requires careful tuning of the optimization step; incorrect parameter choices may cause divergence or traffic oscillations [27]. However, a universal solution that considers all mentioned factors and ensures stable, fair, and efficient real-time load balancing in microservice

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

clouds has not yet been proposed. This indicates the need for further research: it is necessary to develop new approaches that overcome current limitations and fill existing scientific gaps in the problem of dynamic load balancing.

RESEARCH AIM AND OBJECTIVES

The aim of this research is to develop a decentralized load balancing method for microservice systems based on the theory of non-cooperative games, that takes into account asynchronous updates, limited information availability, and resource heterogeneity.

To achieve this aim, the following objectives were defined:

– to analyze existing approaches to load balancing in distributed and microservice systems, particularly those based on game-theoretic models;

– to identify the limitations of existing solutions related to assumptions of synchrony, centralization, and complete information;

– to construct a formal model of load balancing as a non-cooperative game with a set of agents (services) making independent decisions;

– to develop an adaptive decentralized algorithm for achieving equilibrium that is resistant to asynchronous updates and feedback delays;

– to perform simulation of the proposed approach and compare its efficiency with traditional heuristic methods (RR, LC, etc.);

– to evaluate convergence, stability, scalability, and the impact of instance heterogeneity on the quality of load balancing.

DECENTRALIZED LOAD BALANCING METHOD BASED ON A GAME-THEORETIC MODEL

In this study, we developed a mathematically formalized, decentralized load balancing method called Game-Theoretic Load Balancer (GTLB) which is based on the theory of potential games. The system is considered as a set of rational agents, where each microservice instance acts as a player seeking to minimize its own cost of processing requests in a dynamic environment.

Let there be a set of service instances $E = \{1, 2, \ldots, N\}$.

The total request flow is characterized by an intensity λ, which is distributed among the services through a strategy vector:

$$x = (x_1, x_2, \ldots, x_N),\ x_i \geq 0,\ \sum_{i=1}^{N} x_i = 1,$$

where x_i is the fraction of requests directed to the i-th node. The service throughput is denoted by μ_i, and the load on it is defined as follows:

$$\rho_i = \frac{\lambda x_i}{\mu_i},$$

and the system stability condition requires that $\rho_i<1$ for all $i \in E$.

The average service time is modeled based on the M/M/1 queue:

$$T_i(\rho_i) = \frac{1}{\mu_i - \lambda x_i}.$$

The total cost, reflecting the average response time and the overload effect, is described by the following function:

$$C_i(x) = w_i T_i(\rho_i) + \alpha_i \rho_i^2 = \frac{w_i}{\mu_i - \lambda x_i} + \alpha_i \left(\frac{\lambda x_i}{\mu_i}\right)^2,$$

where $w_i>0$ is the service weight coefficient, and $\alpha_i>0$ determines the degree of nonlinear cost growth when the optimal utilization is exceeded.

Each player seeks to minimize its own cost:

$$x_i^* = \arg \min_{x_i \in [0,1]} C_i(x),$$

given that all players act simultaneously. This defines a non-cooperative game $G=\langle E, \{x_i\}, \{C_i\}\rangle$.

To ensure convergence and the existence of Nash equilibrium, the model is formulated as an exact potential game with the following potential function:

$$\Phi(x) = \sum_{i=0}^{N} \int_0^{x_i} \frac{\partial C_i(s, x_{-i})}{\partial s}\, ds = \sum_{i=0}^{N} \left(\frac{w_i}{\mu_i - \lambda x_i} + \frac{\alpha_i\ \lambda^2\ x_i^2}{\mu_i^2}\right).$$

For any two states x and x′, the following relation holds:

$$C_i(x_i', x_{-i}) - C_i(x_i, x_{-i}) = \Phi(x_i', x_{-i}) - \Phi(x_i, x_{-i}),$$

which guarantees the equivalence between individual and global improvement.

The Nash equilibrium x^* corresponds to the minimum of the potential function:

$$x_i^* = \arg \min_{x_i \in \Delta_N} \Phi(x),$$

where Δ_N is the simplex of strategies. The existence of a minimum follows from the continuity of Φ on the compact set Δ_N.

For practical implementation, a stochastic best-response dynamic in the form of Softmax decoding is applied. At the t-th step, each player evaluates its current cost $C_i^{(t)}$, after which the probability of selecting a strategy is defined as:

$$p_i^{(t+1)} = \frac{\exp(-\beta C_i^{(t)})}{\sum_{j=1}^{N} \exp(-\beta C_i^{(t)})},$$

where β=1/T is the "temperature" parameter that controls the level of stochasticity: as $T \to 0$, the strategy becomes deterministic, while for large T, it approaches a uniform distribution.

The current strategy is updated according to the exponential rule:

$$x_i^{(t+1)} = (1-\eta)x_i^{(t)} + \eta p_i^{(t+1)},$$

where $\eta \in (0,1]$ is the update step. Such a stochastic process approximates the gradient descent of the potential function:

$$\nabla_{x_i} \Phi(x) = \frac{\lambda w_i}{(\mu_i - \lambda x_i)^2} + 2\alpha_i \frac{\lambda^2 x_i}{\mu_i^2},$$

where $\nabla x_i \Phi(x)$ is the partial derivative (gradient) of the potential function Φ with respect to the variable x_i; λ is the total request flow intensity, i.e., the number of requests arriving in the system per unit of time; w_i is the weight coefficient of the i-th service, representing its priority or criticality (larger values indicate higher "cost" of response time for this node); μ_i is the performance or throughput of the i-th service (the average processing rate of requests); x_i is the fraction of requests directed to the i-th node; α_i is the nonlinear penalty coefficient for overload.

In each iteration, the expected change in the potential function is non-positive, which guarantees a monotonic decrease of Φ and the convergence of the system to the stationary distribution x^*. In real time, the GTLB method operates as follows: for each node i, the current CPU utilization $u_i(t)$ is observed, based on which the actual cost function is determined as follows:

$$C_i(t) = \frac{w_i}{\mu_i - \lambda_i(t)} + \alpha_i u_i^2(t).$$

According to the Softmax rule, the probability of routing a request to node i is calculated as follows:

$$p_i(t) = \frac{e^{-\beta C_i(t)}}{\sum_j e^{-\beta C_j(t)}}.$$

Upon each incoming request, the API gateway selects a node with probability $p_i(t)$. Updates occur asynchronously, taking into account a delay τ that models real network conditions.

To prove the stability of the process, a constraint on the update step is introduced:

$$\eta < \frac{2}{L + \tau},$$

where L is the Lipschitz constant of the gradient of the potential function; τ is the update or feedback delay in the system.

This condition ensures that even under asynchronous updates, the system remains stable and deviations from equilibrium decrease exponentially.

Thus, the GTLB method mathematically describes a self-regulating decentralized load balancing process as a stochastic dynamic system that minimizes the potential in the strategy space. It combines the guarantee of the existence of a Nash equilibrium with practical convergence to a stationary state within polynomial time:

$$t_{conv} = \mathrm{O}\,(N^2 \log(1/\varepsilon)),$$

where ε is the predefined precision of achieving equilibrium.

Thus, the decentralized GTLB approach ensures analytical stability, proven convergence, and applicability in real-world asynchronous microservice architectures.

EXPERIMENTAL RESEARCH RESULTS

The experimental part was conducted to compare the efficiency of the proposed GTLB algorithm with classical load balancing mechanisms such as RR and LC. For this purpose, a simulation environment was created using the SimPy library, which provides event-based modeling of computational processes and queues. During the experiments, the number of service instances varied from 5 to 20. The maximum throughput of each instance was μ_i = 100 requests per second. To compute the cost function, the coefficient α_i = 2.0 was used, and for Softmax decoding, the temperature parameter β = 1.5.

The testing was conducted under two load scenarios:

1) stationary load, corresponding to a stable flow of requests;

2) bursty load, characterized by peak spikes up to 80 % overload.

The efficiency evaluation was carried out using three groups of metrics:

– average system response time (T_{avg});

– standard deviation of CPU utilization (σ_{CPU}), reflecting the uniformity of request distribution among instances;

– system stabilization time (t_{stab}), defined as the moment when σ_{CPU} decreased below 3 %.

The results showed that under loads exceeding 60 % of the system's maximum capacity, GTLB significantly outperforms the RR and LC algorithms (Fig. 1). Due to its game-theoretic request

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

distribution model, GTLB limits tail latency and prevents sharp fluctuations in response time.

During sudden traffic surges, GTLB equalizes the load among instances. Its CPU utilization variation is noticeably lower than that of LC, indicating better performance and showing that the system operates near the Nash equilibrium (Fig. 2).

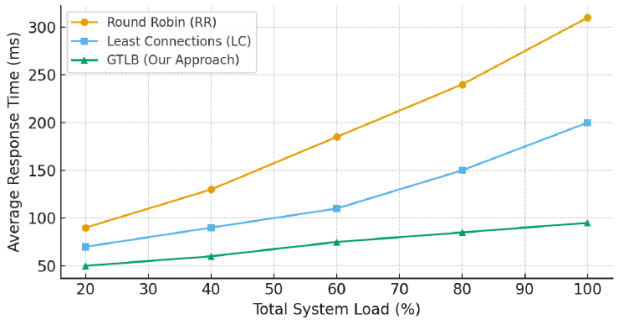

Fig. 1. **Average Response Time vs. Total Load**
***Source*: compiled by the authors**

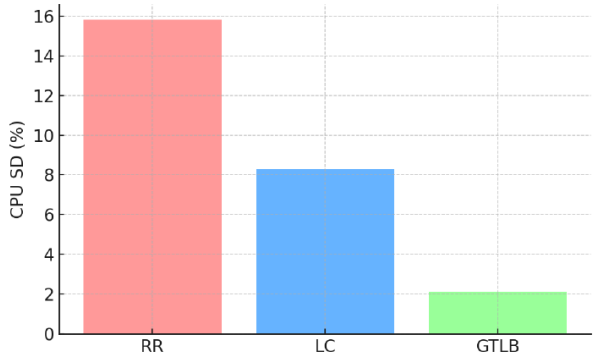

Fig. 2. **CPU Utilization Standard Deviation under Bursty Load**
***Source*: compiled by the authors**

The comparative analysis presented in Table 1 demonstrates the effectiveness of the proposed GTLB method under peak load conditions of 400 requests per second.

Table 1. **Metric Comparison under Peak Load**

Algorithm	Average Response Time (ms)	CPU Standard Deviation (%)
Round Robin (RR)	185.4	15.8
Least Connections (LC)	110.2	8.3
GTLB (proposed)	75.6	2.1

***Source*: compiled by the authors**

The graph below shows how the stabilization time changes with an increasing number of instances (Fig. 3).

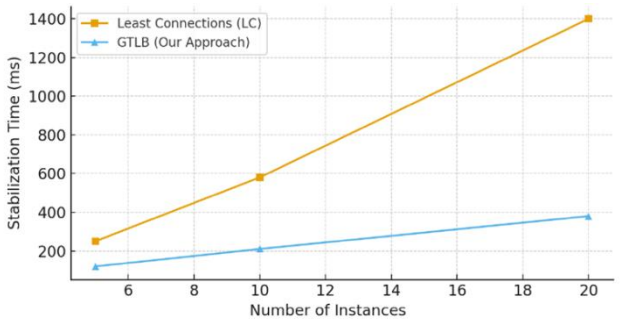

Fig. 3. **Stabilization Time vs. Number of Instances**
***Source*: compiled by the authors**

The experiment also confirmed that the system's stabilization time increases polynomially with the number of instances, remaining several times lower than that of LC (Table 2).

Table 2. **Stabilization time versus system scale**

Number of instances	Stabilization time of LC (ms)	Stabilization time of GTLB (ms)
5	250	120
10	580	210
20	1400	380

***Source*: compiled by the authors**

Thus, GTLB demonstrates stable convergence and lower load variance, ensuring that the system operates near the Nash equilibrium.

CONCLUSIONS

As a result of the conducted research, a game-theoretic method for decentralized load balancing, GTLB, was developed, providing coordinated request distribution in microservice architectures without the need for centralized control. The method formalizes the load balancing problem as an exact potential game, where each service instance acts as a rational agent minimizing its own cost. This enables the achievement of Nash equilibrium, corresponding to a stable system state in which no node has an incentive to unilaterally change its strategy.

The developed stochastic Softmax-update algorithm implements gradient descent of the potential function, taking into account asynchrony and communication delays between agents. It has been proven that the process converges to a stationary state in polynomial time, ensuring scalability in systems with a large number of service instances.

Experimental modeling in the SimPy environment demonstrated a significant advantage of GTLB over classical algorithms such as Round

Robin and Least Connections. The proposed method reduced the average system response time under peak load, decreased CPU utilization variance, and achieved faster stabilization of the operating state.

The obtained results confirm that the proposed method demonstrates analytical stability, proven convergence, and high efficiency in dynamic microservice environments. Its application improves load distribution and enhances system reliability even under sharp fluctuations in request intensity.

REFERENCES

1. Xiao, H., Zhang, Z. & Zhou, Z. “GWS – a collaborative load-balancing algorithm for Internet-of-Things”. *Sensors.* 2018; 18 (8): 2479, https://www.scopus.com/pages/publications/85051077846?origin=resultslist. DOI: https://doi.org/10.3390/s18082479.

2. Bandyopadhyay, A., Swain, S., Singh, R. K., Sarkar, P., Bhattacharyya, S. & Mrsic, L. “Game-theoretic resource allocation and dynamic pricing mechanism in fog computing”. *IEEE Access.* 2024; 12: 51704–51718, https://www.scopus.com/pages/publications/85189629587?origin=resultslist. DOI: https://doi.org/10.1109/ACCESS.2024.3384334.

3. Komleva, N. & Nikitchenko, M. “Method for incremental control of consistency between structural and behavioral views of software architecture”. *Applied Aspects of Information Technology*. 2025; 8 (2): 162–177. DOI: https://doi.org/10.15276/aait.08.2025.11.

4. Komleva, N., Liubchenko, V. & Zinovatna, S. “Evaluation of the quality of survey data and its visualization using dashboards”. *Computer Science and Information Technologies.* 2020; 2: 234–237, https://www.scopus.com/pages/publications/85100509851?origin=resultslist.
DOI: https://doi.org/10.1109/CSIT49958.2020.9321970.

5. Naaz, Z., Joshi, G. & Sharma, V. “Load-balancing model using game theory in edge-based IoT network”. *Pervasive and Mobile Computing.* 2025; 109: 102041, https://www.scopus.com/pages/publications/105000559042?origin=resultslis.t
DOI: https://doi.org/10.1016/j.pmcj.2025.102041.

6. Ulichev, O. & Kulahin, V. “Game-theoretic approach to microservice optimization”. *Central Ukrainian Scientific Bulletin (Technical Sciences).* 2025; 12 (43) Part I: 44–57. DOI: https://doi.org/10.32515/2664-262X.2025.12(43).1.44-57.

7. Telmanov, M., Suchkov, M., Abdiakhmetova, Z. & Kartbayev, A. “Strategic processor task allocation through game-theoretic modeling in distributed computing environments”. *Bulletin of Electrical Engineering and Informatics.* 2025; 14 (2): 1371–1380, https://www.scopus.com/pages/publications/85216537772?origin=resultslist.
DOI: https://doi.org/10.11591/eei.v14i2.9257.

8. Tripathi, R., Sivaraman, V., Tamarapalli, V., Chronopoulos, A. T. & Siar, H. “Non-cooperative power and latency aware load balancing in distributed data centers”. *Journal of Parallel and Distributed Computing.* 2017; 107: 76–86, https://www.scopus.com/pages/publications/85019110726?origin=resultslist. DOI: https://doi.org/10.1016/j.jpdc.2017.04.006.

9. Liu, S., Tian, J., Deng, X., Yuan, Z., Bian, J. et al. “Stackelberg game-based task offloading in vehicular edge computing networks”. *International Journal of Communication Systems.* 2021; 34 (16): e4947. DOI: https://doi.org/10.1002/dac.4947.

10. Abedin, S. F., Bairagi, A. K., Munir, M. S., Tran, N. H.& Hong, C. S. “Fog load balancing for massive machine type communications: a game and transport theoretic approach”. *IEEE Access.* 2019; 7: 4204–4218, https://www.scopus.com/pages/publications/85059274942?origin=resultslist. DOI: https://doi.org/10.1109/ACCESS.2018.2888869.

11. Polgar, Z. A. & Varga, M. “Game theory-based load-balancing algorithms for small cells’ wireless backhaul connections”. *Applied Sciences.* 2023; 13 (3): 1485, https://www.scopus.com/pages/publications/85147871379?origin=resultslist.
DOI: https://doi.org/10.3390/app13031485.

12. Kungurtsev, O. & Komleva, N. “Implementation of Class Interaction under Aggregation Conditions”. *Eastern-European Journal of Enterprise Technologies.* 2024; 2 (2) (128): 20–30, https://www.scopus.com/pages/publications/85194906111?origin=resultslist.
DOI: https://doi.org/10.15587/1729-4061.2024.301011.

13. Surkov, S. S., Martynyuk, O. M., Drozd, O. V. & Drozd, M. O. “A model and method for enhancing the efficiency of processing operation queues at maximum server equipment load.” *Applied Aspects of Information Technology*. 2024; 7 (2): 125–134. DOI: https://doi.org/10.15276/aait.07.2024.9.

14. Yi, C., Cai, J., Zhu, K. & Wang, R. “A queueing game based management framework for fog computing with strategic computing speed control”. *IEEE Transactions on Mobile Computing.* 2022; 21(5): 1537–1551. https://www.scopus.com/pages/publications/85128526920?origin=resultslist. DOI: https://doi.org/10.1109/TMC.2020.3026194.

15. Swathy, R., Vinayagasundaram, B., Rajesh, G., Nayyar, A., Abouhawwash, M. & Abu-Elsoud, M. “Game theoretical approach for load balancing using SGMLB model in cloud environment”. *PLOS ONE*. 2020; 15 (4): e0231708, https://www.scopus.com/pages/publications/85083505307?origin=resultslist. DOI: https://doi.org/10.1371/journal.pone.0231708.

16. Jie, Y., Tang, X., Choo, K.-K. R., Li, M., Guo, C & Su, S. “Online task scheduling for edge computing based on repeated stackelberg game”. *Journal of Parallel and Distributed Computing.* 2018; 122: 159–172, https://www.scopus.com/pages/publications/85052873006?origin=resultslist. DOI: https://doi.org/10.1016/j.jpdc.2018.07.019.

17. He, Q., Wang, H., Jin, H. et al. “A Game-theoretical approach for user allocation in edge computing environment”. *IEEE Transactions on Parallel and Distributed Systems.* 2020; 31 (4): 515–529. https://www.scopus.com/pages/publications/85076917885?origin=resultslist.
DOI: https://doi.org/10.1109/TPDS.2019.2938944.

18. Yuan, X., Min, G., Yang, L. T., Ding, Y. & Fang, Q. “A game theory-based dynamic resource allocation strategy in geo-distributed datacenter clouds”. *Future Generation Computer Systems.* 2017; 76: 63–72, https://www.scopus.com/pages/publications/85020441051?origin=resultslist.
DOI: https://doi.org/10.1016/j.future.2017.04.046.

19. Wang, Y., Wang, J. & Sun, J. “A game-theoretic based resource allocation strategy for cloud computing services”. *Scientific Programming.* 2016. p. 1629893. DOI: https://doi.org/10.1155/2016/1629893.

20. Wu, H., Shi, B., He, Q., Cui, G., Chen, S., Feng, Z., Zomaya, A. Y. & Deng, S. “A game-theoretic approach for microservice request dispatching in mobile edge computing systems”. *IEEE Transactions on Services Computing.* 2025; 18 (5): 2503–2516, https://www.scopus.com/pages/publications/105014779787?origin=resultslist. DOI: https://doi.org/10.1109/TSC.2025.3602905.

21. Kishor, A., Niyogi, R. & Veeravalli, B. “A game-theoretic approach for cost-aware load balancing in distributed systems”. *Future Generation Computer Systems.* 2020; 109: 29–44, https://www.scopus.com/pages/publications/85082017829?origin=resultslist.
DOI: https://doi.org/10.1016/j.future.2020.03.027.

22. Komleva, N. O., Cherneha, K. S., Tymchenko, B. I. & Komlevoy, O. M. “Intellectual Approach Application for Pulmonary Diagnosis”. *Proceedings of the IEEE First International Conference on Data Stream Mining & Processing (DSMP).* 2016; 1: 48–52. https://www.scopus.com/pages/publications/84994235648?origin=resultslist.
DOI: https://doi.org/10.1109/DSMP.2016.7583505.

23. Shuryhin, K. A. & Zinovatna, S. L. “Recommendation system for financial decision-making using Artificial Intelligence.” *Applied Aspects of Information Technology*. 2024; 7 (4): 348–358. DOI: https://doi.org/10.15276/aait.07.2024.24.

24. Siar, H., Kiani, K. & Chronopoulos, A. T. “An effective game theoretic static load balancing applied to distributed computing”. *Cluster Computing.* 2015; 18 (4): 1609–1623, https://www.scopus.com/pages/publications/84983393840?origin=resultslist.
DOI: https://doi.org/10.1007/s10586-015-0486-0.

25. Kumar, S., Sharma, V. & You, I. “A game-theoretic approach for increasing resource utilization in edge computing enabled IoT”. *IEEE Access.* 2022; 10: 57974–57989, https://www.scopus.com/pages/publications/85130435382?origin=resultslist.
DOI: https://doi.org/10.1109/ACCESS.2022.3175850.

26. Proietti Mattia, G., Pietrabissa, A. & Beraldi, R. “A load balancing algorithm for equalising latency across fog or edge computing nodes”. *IEEE Transactions on Services Computing.* 2023; 16 (5): 3129–3140, https://www.scopus.com/pages/publications/85153382886?origin=resultslist.
DOI: https://doi.org/10.1109/TSC.2023.3265883.

27. Niu, Y. et al. “Load Balancing Across Microservices”. *IEEE INFOCOM Conference Proceedings.* 2018. DOI: https://doi.org/10.1109/INFOCOM.2018.8486300.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article

Received 30.09.2025
Received after revision 26.11.2025
Accepted 03.12.2025

DOI: https://doi.org/10.15276/hait.08.2025.31
УДК 004.89:004.272.4:681.518

Ігрово-теоретичний метод децентралізованого балансування навантаження в мікросервісних архітектурах

Грищенко Андрій Ігорович[1)]
ORCID: https://orcid.org/0009-0007-1191-7948; andrew.hryshchenko@gmail.com
Комлева Наталія Олегівна[2)]
ORCID: https://orcid.org/0000-0001-9627-8530, komleva@op.edu.ua. Scopus Author ID: 57191858904
[1)] Компанія Lowe's, Inc., м. Мурсвілл, штат Північна Кароліна, США
[2)] Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, Україна, 65044

АНОТАЦІЯ

У статті представлено теоретико-ігровий метод децентралізованого балансування навантаження в мікросервісних архітектурах, спрямований на підвищення ефективності розподілу запитів між екземплярами сервісів без використання централізованих контролерів. Основна ідея полягає в представленні процесу балансування як некооперативної потенційної гри, в якій кожен мікросервіс розглядається як автономний агент, який прагне мінімізувати власну функцію витрат. На відміну від традиційних алгоритмів, таких як Round Robin та Least Connections, запропонований підхід базується на адаптивному коригуванні стратегій агентів залежно від поточного стану системи, що забезпечує досягнення рівноваги Неша та стабільний розподіл навантаження.

Математична модель теоретико-ігрового методу децентралізованого балансування навантаження враховує інтенсивність потоку запитів, пропускну здатність кожного вузла та квадратичну складову витрат, пов'язаних з перевантаженням ресурсів. Для оптимізації процесу прийняття рішень використовується стохастична динаміка Softmax-update, яка апроксимує градієнтний спуск потенційної функції. Це дозволяє системі поступово балансувати навантаження навіть за наявності асинхронних оновлень та затримок зв'язку між вузлами. Було доведено, що процес сходиться до стаціонарного стану за поліноміальний час, що забезпечує масштабованість та передбачувану поведінку у великих розподілених середовищах.

Експериментальне дослідження, проведене в середовищі моделювання SimPy, продемонструвало, що запропонований метод значно перевершує класичні алгоритми за ключовими показниками. Під час пікового навантаження алгоритм балансування навантаження на основі теорії ігор зменшив середній час відгуку системи порівняно з алгоритмами Round Robin та Least Connections. Стандартне відхилення використання процесора зменшилося, що свідчить про більш збалансований розподіл робочого навантаження та відсутність перевантажених обчислювальних вузлів.

Отримані результати підтверджують аналітичну стійкість, збіжність та практичну ефективність ігрового підходу. Розроблений метод забезпечує адаптивне саморегулювання системи, мінімізує ризик перевантаження та підвищує надійність мікросервісних архітектур. Подальші дослідження повинні бути зосереджені на інтеграції ігрового методу децентралізованого балансування навантаження з хмарними оркестраторами та поширенні моделі на багаторівневі ігри в гібридних обчислювальних середовищах.

Ключові слова: балансування навантаження; мікросервісна архітектура; теорія ігор; рівновага Неша; розподілені системи; програмна інженерія; оптимізація ресурсів

ABOUT THE AUTHORS

Andrii I. Hryshchenko - Senior Software Engineer at Lowe's Companies, Inc., 1000 Lowe's Blvd., Mooresville, NC 28117, USA
ORCID: https://orcid.org/0009-0007-1191-7948; andrew.hryshchenko@gmail.com
***Research field*:** Web Performance Optimization; Distributed Systems; Microservice Architecture; Edge Computing; Reliability Engineering

Грищенко Андрій Ігорович - провідний інженер-програміст компанії Lowe's Companies, Inc. 1000 бульвар Лоуз, м. Мурсвілл, штат Північна Кароліна, 28117, США

Nataliia O. Komleva - PhD, Associate Professor, Head of System Software Department, Odesa Polytechnic National University, Odesa, Ukraine
ORCID: http://orcid.org/0000-0001-9627-8530; komleva@op.edu.ua. Scopus Author ID: 57191858904
Scientific field: Data Analysis, Software Engineering, Distributed Computing

Комлева Наталія Олегівна - кандидат технічних наук, завідувач кафедри Інженерії програмного забезпечення. Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, Україна, 65044