**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
"КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО"
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
"КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО"
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

Кваліфікаційна наукова
праця на правах рукопису

ПРОКОПЕЦЬ НАТАЛІЯ АНДРІЇВНА

УДК 004.75

ДИСЕРТАЦІЯ

ЕНЕРГОЕФЕКТИВНЕ ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ МЕРЕЖІ

Спеціальність 172 – Телекомунікації та радіотехніка

Галузь знань 17 – Електроніка та телекомунікації

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____Прокопець Н.А.

Науковий керівник: Глоба Л.С. доктор технічних наук, професор

Київ – 2022

АНОТАЦІЯ

Прокопець Н.А. Енергоефективне обслуговування навантаження інформаційно-комунікаційної мережі. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 172 – Телекомунікації та радіотехніка. – Навчально-науковий інститут телекомунікаційних систем КПІ ім. Ігоря Сікорського, Київ, 2022.

У дисертаційній роботі розв'язано актуальну науково-практичну задачу підвищення енергоефективності та продуктивності обслуговування навантаження інформаційно-комунікаційної мережі (ІКМ) при виконанні вимог щодо доступності системи обслуговування навантаження за рахунок застосування комплексного методу енергоефективного обслуговування навантаження.

Функціонування сучасної ІКМ великою мірою залежить від програмного забезпечення (ПЗ), що виконує різноманітні мережеві задачі. Це обумовлено розвитком ряду технологій та концепцій, зокрема SDN (Software-Defined Networking), NFV (Network Functions Virtualization), логічного поділу мережі (Network Slicing), периферійних обчислень (Edge Computing) та bDDN (Big data driven networking). Задачі, що вирішуються в рамках цих концепцій, формують обчислювальне навантаження, для обслуговування якого необхідною є побудова та підтримка розподілених обчислювальних систем як невід'ємної частини архітектури ІКМ. При цьому, особливості цих типів навантаження формують специфічні вимоги щодо його обслуговування. Проведений у роботі аналіз вимог різних типів навантаження ІКМ згідно з рекомендаціями Міжнародної спілки електрозв'язку дозволив визначити основні показники ефективності системи розподіленого обслуговування навантаження у складі ІКМ та серверного кластера як одиниці розподіленого центру обробки даних (ЦОД) у складі ІКМ зокрема: показники енергоефективності E_{Σ} та продуктивності C_{Σ} обробки обчислювального навантаження, а також коефіцієнт готовності системи розподіленого обслуговування навантаження p_{avail}. На основі цих показників сформовано

критерій оптимальності процесу обслуговування навантаження в інформаційно-комунікаційній мережі K_{opt}.

В ході аналізу існуючих підходів щодо підвищення енергоефективності розподіленого обслуговування навантаження виявлено певні їх недоліки, а саме: статичні підходи не враховують динамічну змінюваність інтенсивності навантаження; динамічні підходи, що застосовуються на рівні апаратного забезпечення мають високу складність та вартість впровадження. Серед відомих динамічних підходів, що використовуються на рівні програмного забезпечення, підходи щодо консолідації та масштабування обчислювальних ресурсів не враховують показник доступності системи, можуть негативно впливати на продуктивність системи, особливо у випадку динамічних змін інтенсивності навантаження, не використовують індивідуальні характеристики енергоспоживання обчислювальних вузлів, що призводить до неоптимального використання обчислювальних ресурсів. Серед методів енергоефективного розподілу навантаження було відзначено алгоритм планування навантаження Backfill, основною перевагою якого є мінімізація простою обчислювальних вузлів за рахунок щільного розподілу обчислювальних робіт. Однак ефективність цього підходу значно зменшується у випадку невисокої інтенсивності вхідного навантаження, крім того, він не враховує індивідуальні характеристики енергоспоживання та продуктивності обчислювальних вузлів. Окремим сукупним недоліком існуючих підходів є те, що кожен з них вирішує задачу підвищення енергоефективності обслуговування навантаження з урахуванням лише частини аспектів цього процесу та показників його ефективності, що визначило необхідність систематизації та формалізації процесу обслуговування навантаження в інформаційно-комунікаційній мережі.

З метою систематизації та формалізації процесу обслуговування навантаження ІКМ як об’єкта дослідження побудовано онтологічну модель досліджуваної системи розподіленого обслуговування навантаження. Це дозволило якісно описати складні взаємозв’язки між виділеними показниками ефективності

досліджуваного процесу та параметрами, що впливають на них.

З метою отримання кількісної оцінки взаємозв'язків між показниками ефективності досліджуваного процесу та параметрами, що на них впливають, побудовано математичну модель системи розподіленого обслуговування навантаження у складі ІКМ як системи масового обслуговування (СМО). У процесі побудови моделі запропоновано метод переходу від нестаціонарного неординарного вхідного потоку заявок до стаціонарного ординарного потоку шляхом дискретизації кривої інтенсивності вхідного навантаження та за допомогою переходу до комплектів серверів, що дозволило значно спростити розрахунки при допустимих втратах точності моделі. Для дискретизації кривої інтенсивності вхідного навантаження запропоновано використання методу квантування за рівнями, що дозволило узгодити величину кроку дискретизації функції зі швидкістю зміни інтенсивності вхідного навантаження. Для визначення кроку квантування запропоновано метод розрахунку порогових величин інтенсивностей вхідного навантаження як функцій кількості обчислювальних вузлів у системі. На основі побудованої математичної моделі запропоновано метод розрахунку шаблонів горизонтального масштабування, що дозволяє визначати оптимальну кількість активних обчислювальних вузлів у кластерах ЦОД ІКМ на кожному інтервалі часу, який визначається швидкістю зміни інтенсивності вхідного навантаження.

Проаналізовано способи визначення індивідуальних моделей енергоспоживання обчислювальних вузлів розподілених ЦОД та обґрунтовано доцільність їх використання у процесі обслуговування навантаження ІКМ. Докладно розглянуто два способи визначення моделей енергоспоживання: емпіричний та програмний. Перший спосіб базується на безпосередньому вимірюванні енергоспоживання вузлів та подальшій інтерполяції отриманих залежностей поліномом ступеня n з метою отримання аналітичних функцій. Другий спосіб базується на програмній оцінці енергетичних моделей з подальшою інтерполяцією отриманих функцій. Рекомендовано використання

методу емпіричного визначення математичних моделей енергоспоживання для нових систем на етапі їх налаштування. У випадку введення нових вузлів до системи або під час її переатестації, рекомендовано використання аналітичного методу визначення моделей енергоспоживання.

Побудовану математичну модель системи у вигляді СМО та розглянуті способи визначення індивідуальних моделей енергоспоживання обчислювальних вузлів покладено в основу нового комплексного методу енергоефективного обслуговування навантаження в ІКМ. Запропонований комплексний метод відрізняється від відомих використанням індивідуальних моделей енергоспоживання обчислювальних вузлів, поєднанням переваг підходів горизонтального масштабування та енергоефективного розподілу задач, врахуванням непередбачуваних динамічних змін інтенсивності вхідного навантаження, що дозволило підвищити енергоефективність процесу обслуговування навантаження без втрати продуктивності та за умови дотримання вимог щодо доступності системи. В рамках запропонованого комплексного методу удосконалено існуючі підходи щодо горизонтального масштабування обчислювальної системи шляхом використання індивідуальних моделей енергоспоживання обчислювальних вузлів та застосування механізму прогнозування динамічних відхилень вхідного навантаження, що дозволило забезпечити інтенсивніше використання найбільш енергоефективного обладнання та вчасно реагувати на непередбачувані зміни інтенсивності вхідного навантаження.

На основі запропонованого комплексного методу енергоефективного обслуговування навантаження створено програмне забезпечення (ПЗ) керування обчислювальними ресурсами в ІКМ, яке дозволяє підвищити енергоефективність та продуктивність розподіленого обслуговування навантаження з дотриманням вимог щодо доступності системи обслуговування та може бути використано для підвищення енергоефективності та продуктивності обробки навантаження у периферійній та центральній хмарі в архітектурі мережі 5G.

Ефективність запропонованого комплексного методу та ПЗ на його основі

перевірено із використанням методів лабораторного експерименту та імітаційного моделювання. Шляхом лабораторного експерименту перевірено ефективність методу у невеликому серверному кластері з 4 обчислювальних вузлів. Імітаційна модель, адекватність якої доведено із використанням критерія Фішера, довела ефективність запропонованого комплексного методу у більшій розподіленій системі із 20 вузлів. Виграш запропонованого комплексного методу у порівнянні із відомими підходами Backfill та Round Robin за показником енергоефективності при цьому склав 9,953% та 26,382% відповідно. Виграш за показником продуктивності становив 5,593% та 49,458% відповідно. При цьому запропонований комплексний метод забезпечує виконання вимог щодо доступності обчислювальних вузлів розподілених ЦОД та дає виграш за обраним критерієм оптимальності на 15,722% у порівнянні із Backfill та на 88,887% у порівнянні з Round Robin, що доводить практичну цінність отриманих результатів дослідження.

Ключові слова: Енергоефективність, продуктивність, доступність системи, інформаційно-комунікаційна мережа, обслуговування навантаження.

ABSTRACT

N.A. Prokopets Energy-efficient processing of the information and communication network workload. – Qualifying scientific work on manuscript rights.

Thesis for graduation scientific degree of Philosophy Doctor by specialty 172 – Telecommunications and radio engineering. – Educational and Scientific Institute of Telecommunication Systems of KPI named after Igor Sikorsky, Kyiv, 2022.

In the thesis, the important scientific and practical problem of increasing the energy efficiency and performance of workload processing in information and communication network (ICN) while meeting the requirements for the availability of the workload processing system was solved through the use of a comprehensive method of energy-efficient workload processing.

The functioning of a modern ICN largely depends on the software that performs various network tasks. This is due to the development of a number of technologies and concepts, including SDN (Software-Defined Networking), NFV (Network Functions Virtualization), Network Slicing, Edge Computing and bDDN (Big data driven networking). The tasks being solved within these concepts form a computing workload, for the processing of which it is necessary to build and maintain distributed computing systems as an integral part of the ICN architecture. At the same time, the peculiarities of these types of workload form specific requirements for its processing. The requirements' analysis conducted for each of these workload types in accordance with the recommendations of the International Telecommunication Union allowed to determine the main performance indicators of the distributed workload processing system as part of the ICN and the server cluster as a unit of the distributed data center as part of the ICN, in particular: energy efficiency E_{Σ} and performance C_{Σ} indicators of workload processing, as well as the system availability factor p_{avail}. Based on these indicators, an optimality criterion of workload processing in ICN K_{opt} was proposed.

During the analysis of the existing approaches to increase the energy efficiency of distributed workload processing, some shortcomings were revealed, namely: static approaches do not take into account the dynamic variability of the workload; dynamic approaches applied at the hardware level have high complexity and cost of

implementation. Among the known dynamic approaches used at the software level, the approaches to consolidation and scaling of computing resources do not take into account the system availability indicator, which can negatively affect the system performance, especially in the case of dynamic changes in the workload arrival rate. They also do not use the individual characteristics of computing nodes' energy consumption, which leads to suboptimal use of computing resources. Among the approaches to energy-efficient workload scheduling, the Backfill workload scheduling algorithm was noted, the main advantage of which is minimizing the downtime of computing nodes due to the dense distribution of computing work. However, the effectiveness of this approach is significantly reduced in the case of a low input workload arrival rate, in addition, it does not take into account the individual characteristics of energy consumption and performance of computing nodes. A separate collective disadvantage of the existing approaches is that each of them solves the problem of increasing the energy efficiency of workload processing taking into account only part of the aspects of this process and its efficiency indicators, which determined the need to systematize and formalize the workload processing process in the information and communication network.

In order to systematize and formalize the workload processing process in the information and communication network as an object of research, an ontological model of a distributed workload processing system was built. This made it possible to qualitatively describe the complex relationships between the selected efficiency indicators of the process being researched and the parameters affecting them.

In order to obtain a quantitative assessment of the relationships between the defined efficiency indicators and the parameters that affect them, a mathematical model of the distributed workload processing system within the ICN as a queuing system (QS) was built. While building the model, a method of transition from a non-stationary non-ordinary input requests flow to a stationary ordinary flow was proposed by discretizing the intensity curve of the input workload and using the transition to sets of servers, which made it possible to significantly simplify calculations with permissible losses of model accuracy. For the discretization of the input workload arrival rate curve, the use of the quantization by levels was proposed, which made it possible to match the size of the

discretization step with the rate of change of the input workload arrival rate. To determine the quantization step, a method of calculating threshold values of input workload arrival rate as a function of the number of computing nodes in the system is proposed. Based on the constructed mathematical model, a method for calculating horizontal scaling patterns is proposed, which allows determining the optimal number of active computing nodes in the system at each time interval, which is determined by the rate the input workload arrival rate change.

The methods of determining individual energy consumption models of computing nodes were analyzed and the expediency of their use in the workload processing process in ICN was substantiated. Two methods of determining energy consumption models were considered in detail: empirical and software-based methods. The first method is based on the direct measurement of the energy consumption of the nodes and further interpolation of the obtained dependencies by a polynomial of a degree n in order to obtain analytical functions. The second method is based on software-based evaluation of energy consumption models with subsequent interpolation of the obtained functions. It is recommended to use the method of empirical energy consumption models determination for new systems at the stage of their configuration. In the case of introducing new nodes to the system or during its re-configuration, it is recommended to use a software-based method for determining energy consumption models.

The built mathematical model of the system in the form of QS and the considered methods of determining individual energy consumption models of computing nodes became the basis of a new comprehensive method of energy-efficient workload processing in computing nodes of distributed data centers. The proposed comprehensive method differs from known ones in the use of individual models of computing nodes' energy consumption, a combination of the advantages of horizontal scaling approaches and energy-efficient scheduling, while taking into account dynamic changes in the input workload arrival rate, which made it possible to increase the energy efficiency of the workload processing without loss of performance and subject to compliance with system availability requirements. As part of the proposed comprehensive method, the existing

approaches to horizontal scaling of the computer system were improved by using individual models of computer nodes' energy consumption and mechanism for predicting dynamic deviations of the input workload arrival rate, which made it possible to ensure more intensive use of the most energy-efficient equipment and to respond in time to unpredictable changes in the input workload arrival rate.

On the basis of the proposed comprehensive method of energy-efficient workload processing, software for managing computing resources has been created, which allows to increase the energy efficiency and performance of distributed workload processing while complying with the requirements for system availability, and can be used to increase the energy efficiency and performance of workload processing in edge and central cloud within the 5G network architecture.

The effectiveness of the proposed comprehensive method and the software based on it was verified using the methods of laboratory experiment and simulation modeling. The effectiveness of the method was tested in a small server cluster with 4 computing nodes by means of the experiment. The simulation model, the adequacy of which was proven using Fisher's test, proved the effectiveness of the proposed comprehensive method in a larger distributed system with 20 nodes. The performance of the proposed comprehensive method in comparison with the known Backfill and Round Robin approaches in terms of energy efficiency was 9.953% and 26.382%, respectively. The performance gain was 5.593% and 49.458% respectively. At the same time, the proposed comprehensive method ensures the fulfillment of the requirements regarding the system availability and gives a gain according to the proposed optimality criterion by 15.722% in comparison with Backfill and by 88.887% in comparison with Round Robin, which proves the practical value of the obtained research results.

Keywords: Energy efficiency, performance, system availability, information and communication network, workload processing.

СПИСОК ПУБЛІКАЦІЙ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Globa, Larysa, Gvozdetska (*Prokopets*), Nataliia & Novogrudska, Rina. (2021). Ontological model for data processing organization in information and communication networks. *System research and information technologies*. pp. 47-60. ISSN: 1681–6048. DOI: https://doi.org/10.20535/SRIT.2308-8893.2021.1.04.

2. Прокопець, Н. А., Глоба, Л. С. (2022). Математичне моделювання процесу обслуговування навантаження в інформаційно-комунікаційній мережі, *Проблеми телекомунікацій*, (1 (30)), сс. 18-31. ISSN: 2220-6922. DOI: https://doi.org/10.30837/pt.2022.1.02.

3. Prokopets, N., Globa, L. (2022). Comprehensive method of energy-efficient workload processing in the information and communication network. *Information and Telecommunication Sciences*, (2), pp. 34-45. ISSN: 2312-4121. DOI: https://doi.org/10.20535/2411-2976.22022.34-45.

4. Глоба, Л. С., Гвоздецька (*Прокопець*), Н. А., Прокопець, В. А., & Степурін, О. В. (2017). Енергоефективний підхід до розподілу задач у серверному кластері. *Вісник Харківського національного університету імені В.Н. Каразіна*. Серія: Математичне моделювання. Інформаційні технології. Автоматизовані системи управління, (34), сс. 18-28. ISSN: 2524-2601.

5. Globa, L., Gvozdetska (*Prokopets*), N., Prokopets, V., Stryzhak, O. (2019). QoS and Energy Efficiency Improving in Virtualized Mobile Network EPC Based on Load Balancing. In: Pejaś, J., El Fray, I., Hyla, T., Kacprzyk, J. (eds) Advances in Soft and Hard Computing. ACS 2018. *Advances in Intelligent Systems and Computing*, vol 889. Springer, Cham. (pp. 140-149). ISSN: 2194-5357. DOI: https://doi.org/10.1007/978-3-030-03314-9_13

Наукові праці, які засвідчують апробацію матеріалів дисертації

6. Gvozdetska (*Prokopets*), N., Globa, L., & Prokopets, V. (2019). Energy-efficient Backfill-based scheduling approach for Slurm resource manager. In 2019 IEEE 15th International Conference on the Experience of Designing and Application of CAD Systems (CADSM) (pp. 1-5). IEEE. ISSN: 2572-7591, DOI:

https://doi.org/10.1109/CADSM.2019.8779312.

7. Globa, L., Gvozdetska (*Prokopets*), N. (2021). Experimental analysis of PCPB-2: Comprehensive energy-efficient approach to distributed workload processing in communication networks. In 2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-3). IEEE. DOI: https://doi.org/10.1109/BlackSeaCom52164.2021.9527759.

8. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Comprehensive energy efficient approach to workload processing in distributed computing environment. In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-6), DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235010.

9. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Energy Efficient Workload Processing Technology for Distributed Computing Environment. In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-3). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235001.

10. Globa, L., Gvozdetska (*Prokopets*), N. (2019). Energy efficient workload processing in distributed computing environment modeling. In 2019 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 1-6). IEEE., DOI: https://doi.org/10.1109/UkrMiCo47782.2019.9165448.

11. Globa, L. S., Prokopets, V. A., & Gvozdetska (*Prokopets*), N. A. (2018). Prognostic-reactive NFV resource allocation method for implementation in virtualized mobile network EPC of Ukraine. In 2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-5). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom.2018.8433602.

12. Schill, A., Globa, L., Stepurin, O., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2017). Power Consumption and Performance Balance (PCPB) scheduling algorithm for computer cluster. In 2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 1-8). IEEE., DOI: https://doi.org/10.1109/UkrMiCo.2017.8095365.

13. Gvozdetska (*Prokopets*), N., Stepurin, O., & Globa, L. (2017). Experimental

analysis of PCPB scheduling algorithm. In 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM) (pp. 244-247). IEEE., DOI: https://doi.org/10.1109/CADSM.2017.7916126.

14. Globa, L., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2021). Providing Energy-efficient and High-performance Infrastructure for Smart Network. In 2021 IEEE International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 133-136). IEEE., DOI: https://doi.org/10.1109/UkrMiCo52950.2021.9716620.

Наукові праці, які додатково відображають наукові результати дисертації

15. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Оптимізація обчислень у дата-центрі за критерієм енергоефективності. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 127-150.

16. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Динамічний розподіл ресурсів у віртуалізованій мережі мобільного оператора України з підвищенням енергоефективності обробки даних. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 151-176.

17. Л.С. Глоба, Н.А. Гвоздецька (*Прокопець*), В.А. Прокопець, & О.В. Степурін (2017). Підхід до енергоефективного планування задач у серверному кластері для оптимізації обробки трафіка передачі даних. *Наукоемкие технологии в инфокоммуникациях: обработка информации, кибербезопасность, информационная борьба : розділ монографії*. Під загальною редакцією В. М. Безрука, В. В. Баранника. Х. : Издательство «Лидер», 2017. сс. 34-51.

18. L.Globa, O. Stryzhak, N. Gvozdetska (*Prokopets*), & V. Prokopets. (2019). Intelligent Workload Scheduling in Distributed Computing Environment for Balance Between Energy Efficiency and Performance. *Scheduling Problems - New Applications*

and Trends. London, 2019. (IntechOpen). 154 с. DOI: http://dx.doi.org/10.5772/intechopen.86874

19. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба. (2017). Підхід до оптимізації обробки даних з урахуванням параметрів енергоефективності та продуктивності. *Одинадцята міжнародна науково-технічна конференція "Проблеми телекомунікацій"*. 2017. сс. 464–466.

ЗМІСТ

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ 18
ВСТУП 19
РОЗДІЛ 1 ПРОБЛЕМА ЕНЕРГОЕФЕКТИВНОСТІ ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ МЕРЕЖІ 26
1.1 Процес обслуговування навантаження інформаційно-комунікаційної мережі 26
1.1.1 Особливості навантаження сучасної інформаційно-комунікаційної мережі 26
1.1.1.1 Вимоги щодо обслуговування та проблема енергоефективності обслуговування різних типів навантаження 33
1.1.2 Характеристика процесу обслуговування навантаження 40
1.1.2.1 Система розподіленого обслуговування навантаження у складі інформаційно-комунікаційної мережі та її параметри 40
1.1.2.2 Показники та критерій ефективності процесу обслуговування навантаження 48
1.2 Існуючі підходи щодо підвищення енергоефективності процесу обслуговування навантаження в інформаційно-комунікаційній мережі 54
1.2.1 Підходи на рівні апаратного забезпечення 56
1.2.2 Горизонтальне масштабування та консолідація 59
1.2.3 Енергоефективний розподіл навантаження 62
ВИСНОВКИ 68
РОЗДІЛ 2 МЕТОДИ ДОСЛІДЖЕННЯ ПРОБЛЕМИ ЕНЕРГОЕФЕКТИВНОСТІ ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ 70
2.1 Формалізація процесу обслуговування навантаження в інформаційно-комунікаційній мережі 71
2.1.1 Аналіз існуючих підходів щодо формалізації складних систем та процесів 71
2.1.2 Онтологічна модель процесу обслуговування навантаження 72
2.2 Математична модель системи обслуговування навантаження як системи масового обслуговування 80
2.2.1 Аналіз існуючих математичних моделей систем розподіленого обслуговування навантаження 80
2.2.2 Математична модель досліджуваної системи 83

2.2.2.1 Метод перетворення моделі з метою спрощення розрахунків..... 92

2.2.2.2 Метод дискретизації функції інтенсивності надходження вхідного навантаження..... 99

2.3 Математична модель енергоспоживання обчислювальних вузлів..... 104

2.3.1 Метод визначення моделей енергоспоживання емпіричним шляхом 105

2.3.1.1 Апроксимація із застосуванням методу найменших квадратів 106

2.3.1.2 Метод інтерполяції 108

2.3.2 Метод визначення моделей енергоспоживання програмним шляхом 111

ВИСНОВКИ..... 114

РОЗДІЛ 3 КОМПЛЕКСНИЙ МЕТОД ЕНЕРГОЕФЕКТИВНОГО ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ МЕРЕЖІ 116

3.1 Комплексний метод енергоефективного обслуговування навантаження 116

3.1.1 Атестація та сортування обчислювальних вузлів..... 119

3.1.2 Метод визначення шаблонів горизонтального масштабування..... 122

3.1.3 Застосування визначених шаблонів для горизонтального масштабування та енергоефективного планування задач 128

3.1.4 Виявлення відхилення поточного навантаження від статистичного та динамічна адаптація шаблонів масштабування..... 131

ВИСНОВКИ..... 137

РОЗДІЛ 4 АНАЛІЗ ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО КОМПЛЕКСНОГО МЕТОДУ ЕНЕРГОЕФЕКТИВНОГО ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ..... 138

4.1 Опис розроблених програмних засобів керування обчислювальними ресурсами 138

4.2 Застосування розроблених програмних засобів в інформаційно-комунікаційній мережі..... 140

4.3 Оцінка ефективності функціонування розробленого програмного забезпечення керування обчислювальними ресурсами 142

4.3.1 Експериментальне дослідження ефективності запропонованого комплексного методу..... 143

4.3.1.1 Постановка експерименту..... 143

4.3.1.2 Оцінка ефективності запропонованого комплексного методу 149

4.3.1.1 Оцінка ефективності енергоефективного алгоритму планування Backfill (порівняння із запропонованим комплексним методом)..... 154

4.3.1.2 Оцінка ефективності алгоритму планування Round Robin (порівняння із запропонованим комплексним методом)..... 158

4.3.1.3 Аналіз результатів експериментальної оцінки ефективності запропонованого комплексного методу та його порівняння з існуючими підходами..... 161

4.3.2 Імітаційне моделювання процесу обслуговування навантаження..... 164

4.3.2.1 Первинний збір інформації та постановка задачі моделювання... 165

4.3.2.2 Створення моделі 170

4.3.2.3 Перевірка адекватності моделі шляхом моделювання процесу обробки вхідного навантаження в умовах аналогічних лабораторному експерименту..... 171

4.3.2.4 Проведення моделювання для серверного кластера з 20 вузлів та обробка отриманих результатів..... 175

4.3.2.5 Аналіз результатів моделювання 182

ВИСНОВКИ..... 185

ЗАГАЛЬНІ ВИСНОВКИ 187

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

АЗ	Апаратне забезпечення
ВМ	Віртуальна машина
ЕОМ	Електронно-обчислювальна машина
ІКМ	Інформаційно-комунікаційна мережа
ІКТ	Інформаційно-комунікаційні технології
ПЗ	Програмне забезпечення
СМО	Система масового обслуговування
ЦОД	Центр обробки даних
ЦП	Центральний процесор
bDDN	Big data driven networking
DVFS	Dynamic Voltage and Frequency Scaling
FCFS	First Come First Served
FLOP	Floating Point Operations
KPI	Key Performance Indicator
NFV	Network Functions Virtualization
PCPB	Power Consumption and Performance Balance
QCI	QoS Class Identifier
QoS	Quality of Service
RAM	Random-Access Memory
RR	Round Robin
SDN	Software-Defined Networking
SLA	Service Level Agreement
SSH	Secure Shell
TCP	Transmission Control Protocol

ВСТУП

Актуальність теми. Функціонування сучасних інформаційно-комунцікаційних мереж (далі ІКМ) передбачає активне залучення технологій та концепцій, що базуються на розв'язанні мережевих задач за допомогою програмного забезпечення. Це зокрема:

1. технології NFV та SDN;
2. концепція логічного поділу мережі (Network Slicing);
3. концепція периферійних обчислень (Edge Computing);
4. концепція мережі, керованої Великими даними (bDDN).

Задачі, що вирішуються за допомогою програмного забезпечення у рамках цих технологій та концепцій формують обчислювальне навантаження, що потребує обслуговування згідно з вимогами, обумовленими особливостями цих задач. Це викликає необхідність побудови та підтримки системи розподіленого обслуговування навантаження у складі ІКМ, що фізично представлена центрами обробки даних (далі ЦОД).

Функціонування інфраструктури територіально розподілених ЦОД є енерговитратним та вносить суттєвий вклад у загальне енергоспоживання сучасних ІКМ. У зв'язку з цим особливої актуальності набуває вирішення проблемних питань із забезпечення енергоефективності розподілених ЦОД. За даними досліджень [1, 2], частка споживання електроенергії такими системами станом на 2020 рік становила близько 1% усієї електроенергії, що виробляється у світі. Результати аналізу стану питання і прогнозу подальшого розвитку розподілених ЦОД показують, що ця частка енерговитрат продовжує швидко зростати, що обумовлено стрімким нарощуванням потужностей обчислювального обладнання ІКМ. При цьому, система обслуговування навантаження у складі ІКМ повинна забезпечувати достатню обчислювальну продуктивність і рівень доступності системи, щоб гарантувати виконання вимог до якості надання послуг для кожного з типів навантаження сучасних ІКМ (навантаження SDN, NFV, Edge Computing, Network Slicing та bDDN).

На сьогоднішній день існує багато наукових і практичних підходів щодо підвищення енергоефективності обслуговування навантаження у розподілених ЦОД [3 - 33]. Ці підходи здебільшого мають на меті зниження енергоспоживання системи при збереженні рівня її продуктивності. Однак, вони не враховують специфіку навантаження ІКМ та жорсткі вимоги щодо доступності системи при його обслуговуванні, що обумовлені необхідністю забезпечення якості послуг “з кінця в кінець” при динамічно змінюваному навантаженні. Так, наприклад, підвищення енергоефективності за рахунок зменшення кількості активних обчислювальних вузлів розподіленої системи може негативно вплинути на якість телекомунікаційних послуг “з кінця в кінець” при динамічно змінюваному навантаженні на систему. Крім того, існуючі підходи не використовують індивідуальні характеристики енергоспоживання обчислювальних вузлів. Однак, їх використання є особливо важливим у гетерогенних системах, де потужність споживання окремих вузлів може суттєво відрізнятись.

Таким чином, актуальним науково-практичним завданням є вирішення проблемного питання щодо забезпечення енергоефективного та продуктивного обслуговування навантаження сучасної інформаційно-комунікаційної мережі за умови виконання вимог щодо доступності системи при динамічно змінюваному навантаженні та з урахуванням індивідуальних характеристик енергоспоживання вузлів системи.

Зв‘язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконувалась згідно з планом наукових досліджень Кафедри інформаційно-комунікаційних технологій та систем Навчально-наукового інституту телекомунікаційних систем:

1. В рамках держбюджетної теми: 2218п «Гетерогенна мережа збору, передачі та обробки інформації для системи розподіленої генерації MicroGrid» (номер державної реєстрації 0119U001184).

2. В рамках держбюджетної теми: 2313п «Побудова інформаційно-аналітичної платформи для супроводження функціонування кіберфізичних систем» (номер державної реєстрації 0120U102298).

3. В рамках Міжнародного проекту “Research Guidance System for Organizations Based on Self-improvement Domain Knowledge Map and Customized Industry Knowledge Graph” за співпраці НТУУ «КПІ ім. І. Сікорського» та Інституту інформаційних досліджень академії наук провінції Шандунь, КНР (номер договору 0305/55-М).

Мета і задачі дослідження. Метою дисертаційної роботи є підвищення енергоефективності та продуктивності обслуговування навантаження інформаційно-комунікаційної мережі при дотриманні вимог щодо доступності системи за рахунок використання індивідуальних моделей енергоспоживання обчислювальних вузлів, поєднання переваг підходів горизонтального масштабування та енергоефективного розподілу задач з врахуванням непередбачуваних динамічних змін інтенсивності вхідного навантаження.

Для досягнення мети дослідження було поставлено та вирішено такі задачі:

1. Визначити основні характеристики, властивості, показники та критерій ефективності процесу розподіленого обслуговування навантаження інформаційно-комунікаційної мережі. Систематизувати та формально описати складні взаємозв’язки між показниками ефективності досліджуваного процесу та факторами, що впливають на них.

2. Побудувати математичну модель системи розподіленого обслуговування навантаження як системи масового обслуговування (СМО).

3. Запропонувати метод обслуговування навантаження в інформаційно-комунікаційній мережі, який дозволить підвищити енергоефективність та продуктивність процесу обслуговування навантаження за умови дотримання вимог щодо доступності системи.

4. Удосконалити існуючі підходи щодо горизонтального масштабування розподілених обчислювальних систем з метою забезпечення мінімізації простою найбільш ефективного обладнання.

5. На основі запропонованих моделей та методів створити програмне забезпечення керування обчислювальними ресурсами в інформаційно-комунікаційній мережі, а також здійснити його практичне впровадження.

Об'єкт дослідження – процес обслуговування навантаження інформаційно-комунікаційної мережі.

Предмет дослідження – формалізовані моделі та методи оцінювання та підвищення енергоефективності та продуктивності обслуговування навантаження інформаційно-комунікаційної мережі.

Методи дослідження, застосовані для вирішення поставлених завдань:

1. Методи інтелектуального аналізу даних – для аналізу об'єкта дослідження, його формалізації, виділення сутностей та структурних взаємозв'язків у досліджуваній системі.

2. Методи теорії систем масового обслуговування, лінійної екстраполяції, інтерполяції поліномом – для визначення індивідуальних моделей енергоспоживання вузлів обчислювальної системи, для реалізації методів розподіленої енергоефективного обслуговування навантаження (а саме, визначення оптимальних шаблонів масштабування системи, прогнозування та планування навантаження).

3. Методи імітаційного моделювання та лабораторного експерименту – для перевірки ефективності застосування запропонованих математичних моделей та методів та їх порівняння з існуючими аналогами.

Наукова новизна отриманих результатів:

1. Вперше запропоновано комплексний метод енергоефективного обслуговування навантаження інформаційно-комунікаційної мережі, що відрізняється від відомих використанням індивідуальних моделей енергоспоживання обчислювальних вузлів, поєднанням переваг підходів горизонтального масштабування та енергоефективного розподілу задач, врахуванням непередбачуваних динамічних змін інтенсивності вхідного навантаження, що дозволило підвищити енергоефективність та продуктивність процесу обслуговування навантаження за умови дотримання вимог щодо доступності системи.

2. Набули подальшого розвитку підходи щодо формалізації процесу обслуговування навантаження в інформаційно-комунікаційній мережі шляхом

побудови онтологічної моделі процесу обслуговування навантаження, що дозволило систематизувати та якісно описати складні взаємозв'язки між виділеними показниками ефективності досліджуваного процесу та факторами, що впливають на них.

3. Удосконалено математичну модель системи розподіленого обслуговування навантаження інформаційно-комунікаційної мережі як СМО. Запропонована модель відрізняється від відомих тим, що враховує змінний характер інтенсивності вхідного навантаження та можливість використання технік паралелізації при розробці ПЗ сучасних ІКМ. Побудовану математичну модель покладено в основу методу розрахунку шаблонів горизонтального масштабування обчислювальної системи, що дозволило визначати оптимальну кількість активних обчислювальних вузлів у системі на кожному інтервалі часу, який визначається швидкістю зміни інтенсивності вхідного навантаження.

4. Удосконалено існуючі підходи щодо горизонтального масштабування обчислювальної системи шляхом використання індивідуальних моделей енергоспоживання обчислювальних вузлів та застосування механізму прогнозування динамічних відхилень вхідного навантаження, що дозволило забезпечити інтенсивніше використання найбільш енергоефективного обладнання та вчасно реагувати на непередбачувані зміни інтенсивності вхідного навантаження.

Практичне значення одержаних результатів:

1. Усі теоретичні розробки дисертації представлені автором у вигляді комплексного методу енергоефективного обслуговування навантаження, який може бути безпосередньо використаний для підвищення енергоефективності та продуктивності обслуговування навантаження в інформаційно-комунікаційній мережі.

2. На основі розроблених моделей та методів створено програмне забезпечення керування обчислювальними ресурсами, яке дозволяє підвищити енергоефективність та продуктивність розподіленого обслуговування навантаження з дотриманням вимог щодо доступності системи.

3. Виконано практичне впровадження розробленого програмного забезпечення керування обчислювальними ресурсами у компанії ТОВ «Хуавей Україна».

Особистий внесок здобувача. Дисертаційна робота узагальнює результати теоретичних та експериментальних досліджень, проведених автором самостійно. Основні результати, отримані автором особисто: формалізована модель об'єкта дослідження у вигляді онтології [34]; математична модель об'єкта дослідження у вигляді СМО [35]; комплексний метод енергоефективного обслуговування навантаження інформаційно-комунікаційної мережі [36, 37, 38, 39]; метод індивідуальної атестації обчислювальних вузлів та визначення їх моделей енергоспоживання [40, 41, 42, 43]; метод визначення шаблонів горизонтального масштабування на основі моделі системи розподіленої обробки даних як СМО [36, 37, 38]; метод передбачення динамічних змін навантаження та адаптації шаблонів масштабування [44, 45]; імітаційне моделювання роботи запропонованого комплексного методу енергоефективного обслуговування навантаження [46, 47]; експериментальна перевірка ефективності запропонованого комплексного методу енергоефективного обслуговування навантаження [48, 49, 50].

Апробація результатів дисертації. Основні положення та результати дисертаційної роботи були представлені й одержали схвалення на:

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Батумі, Грузія, 2018 р.);

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Одеса, Україна, 2020 р.);

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Бухарест, Румунія, 2021 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Одеса, Україна, 2017 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Одеса, Україна, 2019 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Київ, Україна, 2021 р.);

– Міжнародній науково-технічній конференції «The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)» (Поляна, Свалява, Україна, 2017 р.);

– Міжнародній науково-технічній конференції «The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)» (Поляна, Свалява, Україна, 2019 р.);

– Міжнародній науково-технічній конференції «International Conference on Advanced Computer Systems» (м. Мендзиздроє, Польща, 2018 р.);

– Міжнародній науково-технічній конференції «Проблеми телекомунікацій» (м. Київ, Україна, 2017 р).

Публікації. За результатами досліджень опубліковано 19 наукових праць, у тому числі 3 статті у наукових фахових виданнях України за спеціальністю 172 – Телекомунікації та радіотехніка, 2 статті та 9 тез доповідей на конференціях проіндексованих у базах даних WoS та/або Scopus, 4 розділи колективних монографій, 1 тези у матеріалах конференції.

Структура та обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел із 139 найменувань, 5 додатків. Загальний обсяг роботи 217 сторінок, з яких 170 сторінок основного тексту, 15 сторінок використаних джерел та 11 сторінок додатків. Робота містить 41 рисунок, 15 таблиць.

РОЗДІЛ 1
ПРОБЛЕМА ЕНЕРГОЕФЕКТИВНОСТІ ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ МЕРЕЖІ

1.1 Процес обслуговування навантаження інформаційно-комунікаційної мережі

У цьому підрозділі проаналізовано особливості та вимоги навантаження сучасної інформаційно-комунікаційної мережі, наведено визначення та основні поняття системи розподіленого обслуговування такого навантаження, надано характеристику процесу обслуговування навантаження та сформульовано основні показники та критерій ефективності обслуговування навантаження інформаційно-комунікаційної мережі.

1.1.1 Особливості навантаження сучасної інформаційно-комунікаційної мережі

Ще минулого десятиріччя під поняттям “телекомунікаційної мережі” перш за все мався на увазі комплекс технічних засобів телекомунікацій та споруд, призначених для маршрутизації, комутації, передавання та/або приймання знаків, сигналів, письмового тексту, зображень і звуків або повідомлень будь-якого роду по радіо, дротових, оптичних чи інших електромагнітних системах між кінцевим обладнанням [52]. Однак, у сучасних телекомунікаційних мережах робота технічних засобів телекомунікацій неможлива без програмного забезпечення. Більш того, левова частка функціоналу технічних засобів заміщується програмним забезпеченням внаслідок конвергенції інформаційних та телекомунікаційних технологій. Тому сьогодні доцільним є вживання терміну ***“інформаційно-комунікаційна мережа” (ІКМ)***, тобто сукупність інформаційних та телекомунікаційних систем, які у процесі обробки інформації діють як єдине ціле.

Яскравим прикладом ІКМ, в якій інформаційні та телекомунікаційні технології функціонують як єдине неподільне ціле, є мережа 5-го покоління 5G. Згідно зі специфікацією мережі 5G [54], при побудові ІКМ 5-го покоління

ключовою парадигмою є ***програмованість*** (англ. programmability) мережі. Програмованість забезпечує гнучку адаптацію мережі на різних рівнях, включаючи інфраструктуру, мережеві функції, послуги та додатки. Програмованість у першу чергу забезпечується технологіями NFV (Network Function Virtualization) та SDN (Software-Defined Networking).

Згідно зі специфікацією Європейського інституту телекомунікаційних стандартів ETSI GR NFV 003 [55] ***NFV*** або ***віртуалізація мережевих функцій*** – це принцип відокремлення мережевих функцій від апаратного забезпечення (АЗ), на якому вони працюють з використанням віртуальної апаратної абстракції. Під «мережевою функцією» мається на увазі функціональний блок у межах мережевої інфраструктури, який має чітко визначені зовнішні інтерфейси та чітко визначену функціональну поведінку. Тобто, основною ідеєю NFV є заміна спеціалізованого мережевого обладнання (наприклад, L2-комутаторів, маршрутизаторів, пристроїв NAT (Network Address Translation), фаєрволів, тощо) програмним забезпеченням – віртуалізованими функціями мережі, – консолідованих на АЗ (серверах) загального призначення [56].

SDN або ***програмно конфігуровані мережі,*** згідно з рекомендацією Міжнародної спілки електрозв'язку ITU-T Y.3300 [57], – це набір прийомів та технік, що дозволяє безпосередньо програмувати, оркеструвати, контролювати та керувати мережевими ресурсами, що полегшує проектування, розгортання та функціонування мережевих служб у динамічний і масштабований спосіб. SDN – це підхід до проектування, реалізації та керування мережами, який розділяє управління мережею (площина управління) та процес керування трафіком (площина даних). Завдяки цьому розділенню адміністрування та управління мережею значно спрощуються, оскільки площина управління обробляє лише інформацію, пов’язану з логічною топологією мережі. Площина даних натомість організовує мережевий трафік відповідно до встановленої в площині управління конфігурації. На відміну від звичайних IP-мереж, функції яких децентралізовані, SDN слідує централізованому підходу [58].

Таким чином, за рахунок поєднання технологій NFV та SDN у сучасній інформаційно-комунікаційній мережі одночасно досягається гнучкість надання сервісів для абонента та спрощення адміністрування мережевих ресурсів для оператора мережі. При цьому, задачі:

– віртуалізованих мережевих функцій (маршрутизація, комутація, перетворення мережевих адрес, тощо);

– логічно централізованого контролю мережевих ресурсів;

– оркестрації мережевих ресурсів;

– підлаштування (динамічної реконфігурації) мережевих ресурсів;

– керування фізичними та віртуалізованими ресурсами мережі;

перекладаються зі спеціалізованого мережевого обладнання на сервери загального призначення. Ці задачі створюють ***навантаження***, що потребує обслуговування за допомогою серверної інфраструктури. Тому, для забезпечення функціонування технологій SDN та NFV виникає необхідність побудови та підтримки ***серверної інфраструктури інформаційно-комунікаційної мережі***.

Іншою ключовою концепцією мережі 5-го покоління є ***логічний поділ мережі (Network Slicing)*** – це технологія поділу мережі на логічні рівні поверх єдиної фізичної інфраструктури з метою одночасного задоволення вимог різних сервісів [59]. Кожен логічний рівень мережі представляє окрему ізольовану систему передачі даних “з кінця в кінець”, побудовану на основі вимог специфічного додатка. Згідно з рекомендацією Міжнародної спілки електрозв'язку ITU-T Y.3156 [60], концепція логічного поділу мережі залучає технології штучного інтелекту з метою забезпечення виконання вимог щодо якості обслуговування для користувачів мережі. Для цього мають вирішуватись зокрема такі задачі:

– розпізнавання та оптимізація шаблонів логічних рівнів шляхом зпівставлення вимог щодо якості обслуговування і параметрів шаблону;

– передбачення трафіку;

– оптимізація ресурсів для логічного рівня;

– аналіз причин та локалізація помилок та позаштатних ситуацій.

Згідно з рекомендацією ITU-T Y.3156 [60] ці задачі мають вирішуватись за допомогою алгоритмів машинного навчання, що в свою чергу також потребує залучення ***серверної інфраструктури*** у складі інформаційно-комунікаційної мережі.

Крім того, в мережі 5G провідне місце займає технологія ***периферійних обчислень (Edge Computing)*** [61], що має на увазі перенесення частини ресурсів обробки та зберігання даних якомога ближче до користувача з метою підвищення пропускної здатності обробки даних та зменшення затримки. Аналіз ринку телекомунікаційних послуг, проведений компанією Ericsson [61, 62] показав, що підтримка хмарної інфраструктури та надання сервісів периферійних обчислень є новою можливістю для розширення бізнесу операторів ІКМ. Згідно з рекомендаціями Міжнародної спілки електрозв'язку ITU-T F.743.12 [63] та ITU-T F.743.10 [64], до задач ІКМ, які доцільно вирішувати за допомогою периферійної серверної інфраструктури, належать зокрема:

– задачі аналізу відео у системах відеоспостереження (попередня обробка даних відеоспостереження, виявлення та аналіз позаштатних ситуацій, аналіз міського трафіку, розпізнавання облич, тощо);

– локальне кешування контенту (наприклад, контенту соціальних мереж);

– задачі тривимірної комп’ютерної симуляції для додатків віртуальної та доповненої реальності.

Окрім забезпечення функціонування новітніх технологій в мережі 5G, центри обробки даних виконують також традиційні функції зберігання та обробки інформації двох типів:

– призначена для користувача інформація, тобто та, яка безпосередньо цікавить кінцевих користувачів мережі (інформаційні ресурси);

– службова інформація, яка допомагає надавати послуги користувачам.

Прикладом інформаційних ресурсів першого типу можуть служити web-портали, на яких розташована різноманітна довідкова та новинна інформація, інформація електронних магазинів, тощо. Ресурсами другого типу є різні системи аутентифікації і авторизації користувачів, за допомогою яких організація, що

володіє мережею, перевіряє права користувачів на отримання тих чи інших послуг; системи білінгу, які в комерційних мережах нараховують плату за отримані послуги; бази даних облікової інформації користувачів, що зберігають імена і паролі, а також переліки послуг, на які підписаний кожен користувач, тощо.

Додатково варто відзначити, що центри обробки та зберігання даних використовуються операторами ІКМ також для аналізу ***Великих даних (Big Data) у мережі*** з метою покращення її технічних та економічних параметрів, персоналізації сервісів, ефективного розподілу коштів. Рекомендація Міжнародної спілки електрозв'язку ITU-T Y.3652 [65] визначає концепцію мережі, керованої Великими даними (bDDN – Big data driven networking) як групу технологій і методів для полегшення адміністрування, обслуговування та оптимізації мережі на основі Великих даних, накопичених мережею. Задачі, що вирішуються у рамках концепції bDDN – це збір та аналіз даних про:

– поточну конфігурацію мережі;

– стан мережевого обладнання;

– статистику мережевого трафіку;

– історію помилок та збоїв у мережі;

– кількість абонентів, що обслуговуються базовими станціями;

– дані білінгу (дані про кількість дзвінків, повідомлень і платежів абонентів), тощо [65, 66].

Таким чином, задачі, що вирішуються сучасними ІКМ потребують наявності ***обчислювальної інфраструктури***, тобто центрів обробки та зберігання даних (ЦОД), як невід'ємної частини ІКМ. При цьому, крім традиційних функцій (авторизація та аутентифікація користувачів, нарахування оплати за послуги, тощо), у сучасних ІКМ на кшталт 5G обчислювальна інфраструктура необхідна для підтримки роботи таких ключових технологій як SDN, NFV та логічний поділ мережі (Network Slicing) з метою забезпечення гнучкості мережі та спрощення керування нею, надання сервісів Edge Computing, а також для аналізу Великих даних з метою покращення якості послуг.

Визначимо місце обчислювальної інфраструктури в архітектурі сучасної ІКМ. Для цього розглянемо загальну структуру, ресурси та принципи побудови сучасної ІКМ на прикладі мережі 5G, проаналізувавши технічну документацію, надану європейською організацією 5G PPP (5G PPP – це державно-приватне партнерство у сфері інфраструктури 5G, спільна ініціатива між Європейською Комісією та європейською ІКТ-галуззю (виробники ІКТ, оператори ІКМ, провайдери послуг та дослідницькі установи) [54].

Згідно документації, в загальній архітектурі мережі 5G виділяють 3 логічних рівні (Рисунок 1.1):

1. рівень сервісів (додатків);
2. рівень мережі;
3. ресурсно-функціональний рівень.

Обмін даними (пов'язаними з абонентами, мережею та базовими ресурсами, екземплярами мережевих фрагментів та послуг, тощо) між усіма системними сутностями усіх рівнів відбувається через загальну платформу збору, обробки, аналізу та розподілу даних.

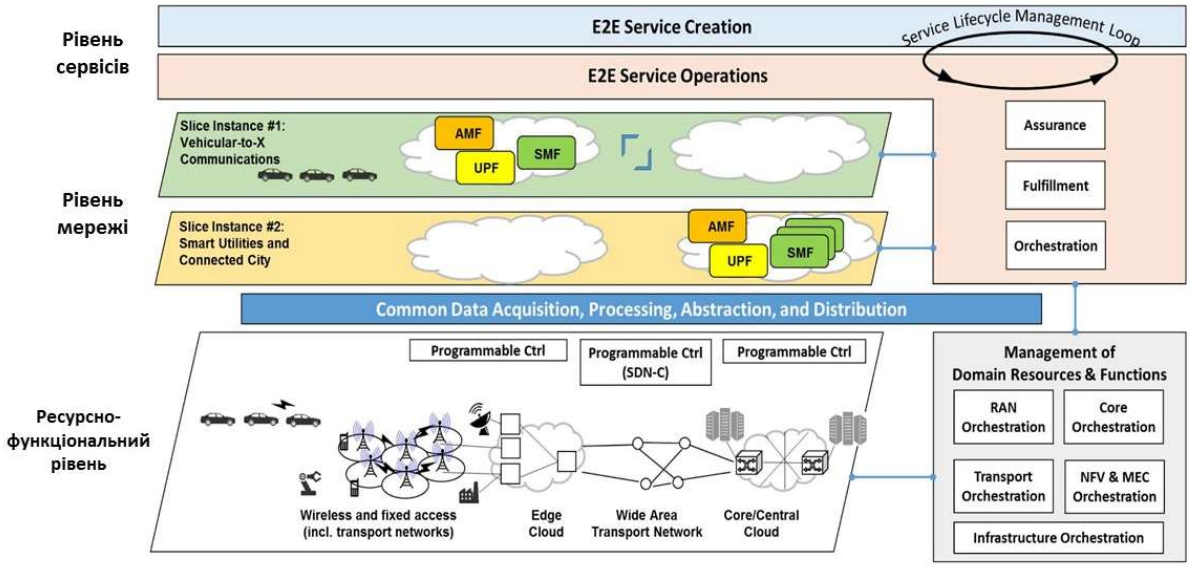


Рисунок 1.1 Загальна архітектура мережі 5G [54]

Рівень сервісів відповідає за керування новими послугами ІКМ. На рівні сервісів здійснюється автоматизація керування життєвим циклом послуг, що

складається з таких фаз: фаза підготовки, створення, активації, фаза активної роботи та фаза виведення з експлуатації.

Рівень мережі відповідає за створення ізольованих наскрізних мережевих фрагментів на одній і тій же фізичній інфраструктурі для їх одночасного використання різнорідними службами. На цьому рівні реалізовані зокрема такі віртуалізовані мережеві функції як:

– AMF (Access and Mobility Management Function) – функція керування доступом та мобільністю;

– SMF (Session Management Function) – функція керування сесіями;

– UPF (User Plane Function) – функція передачі даних користувачів.

Ресурсно-функціональний рівень має на меті створення єдиної мережевої інфраструктури, надання ресурсів для забезпечення функціонування сервісів на вищих рівнях. На ресурсно-функціональному рівні виділяють такі компоненти:

1. мережа доступу (access network);
2. периферійна (гранична) хмара (edge cloud);
3. транспортна мережа або ядро мережі (transport або core network);
4. основна або центральна хмара (core/central cloud).

Мережа доступу – нижній рівень ієрархії ІКМ. До цієї мережі підключаються кінцеві (термінальні) вузли – обладнання, встановлене у користувачів (абонентів, клієнтів) мережі. Наприклад, в разі побудови операторської мережі для надання послуг доступу до мережі Інтернет, кінцевими вузлами можуть є комп'ютери абонентів або абонентські маршрутизатори. Основне призначення мережі доступу – концентрація в порівняно невеликій кількості вузлів базової мережі інформаційних потоків, що надходять по численних каналах зв'язку від обладнання користувачів. ***Периферійна*** або ***гранична хмара*** забезпечує ресурси обчислення та зберігання для додатків в мережі, близькій до кінцевих користувачів. Основні переваги периферійних обчислень включають низьку затримку та велику пропускну здатність для користувача мережі, а також розвантаження основних ресурсів обробки даних. За даними компанії Ericsson [59], розподілена обчислювальна інфраструктура – один з основних функціональних компонентів

периферійної хмари. Крім того, згідно з прогнозами компанії Ericsson [59, 62] підтримка хмарної інфраструктури та надання сервісів Edge Computing є новою можливістю для розширення бізнесу операторів ІКМ. ***Ядро мережі*** об'єднує окремі мережі доступу, виконуючи функції транзиту трафіку між ними по високошвидкісних каналах. Комутатори ядра можуть оперувати не тільки з інформаційними з'єднаннями між окремими користувачами, але і з агрегованими інформаційними потоками, що переносять дані великої кількості призначених для користувача з'єднань. Метою ***основної*** або ***центральної хмари*** є забезпечення роботи віртуалізованих (згідно з технологією NFV [55]) мережевих функцій [54]. Центральна хмара являє собою розподілене обчислювальне середовище, організоване за принципами побудови розподілених обчислювальних систем.

Таким чином, обчислювальна інфраструктура знаходить своє застосування для обслуговування навантаження інформаційно-комунікаційної мережі на кожному з рівнів архітектури мережі 5G:

1. Забезпечують функціонування додатків та аналіз Великих даних на ***рівні сервісів*** (додатків);

2. Підтримують функціонування технології Network Slicing на ***рівні мережі***;

3. Забезпечують функціонування технологій SDN та NFV, здійснення таких необхідних функцій мережі, як авторизація та аутентифікація користувачів, нарахування оплати за послуги, тощо, а також здійснює підтримку технології Edge Computing на ***ресурсно-функціональному рівні***.

1.1.1.1 Вимоги щодо обслуговування та проблема енергоефективності обслуговування різних типів навантаження

Розглянемо вимоги щодо обслуговування кожного зі згаданих типів навантаження інформаційно-комунікаційної мережі. Зокрема звернемо увагу на проблему енергоефективності обслуговування навантаження, якій присвячено серію рекомендацій Міжнародної спілки електрозв'язку ITU-T L.13xx "Довкілля та ІКТ, зміна клімату, електронні відходи, енергоефективність" [68].

Згідно з рекомендацією ITU-T L.1315 [69], що надає високорівневе визначення енергоефективності та вимог до енергоменеджменту для підвищення енергоефективності ІКМ, мета підвищення енергоефективності полягає в реалізації рішень, які забезпечуватимуть таку ж або кращу функціональність, при використанні меншого обсягу електроенергії. У рекомендації енергоефективність EER визначається як $EER = \frac{UsefulWork}{TotalUsedEnerg}$, де $UsefulWork$ – обсяг виконаної пристроєм або ланкою мережі корисної роботи, $TotalUsedEnergy$ – сумарний обсяг спожитої електроенергії.

При цьому зазначається, що з огляду на складність процесів в ІКМ, параметр «обсяг виконаної корисної роботи» не може бути однозначно визначеним для усієї мережі. Натомість він має бути визначеним окремо для пристроїв та підсистем з огляду на їх задачі (наприклад, створення покриття, маршрутизації, обробки даних).

Розглянемо вимоги щодо обслуговування окремих типів навантаження інформаційно-комунікаційної мережі.

Навантаження SDN та NFV

Обслуговування навантаження та визначення енергоефективності цього процесу для SDN та NFV регулюється Міжнародною спілкою електрозв'язку у документах ITU-T L.1360 [70] та ITU-T L.1361 [71]. У цих рекомендаціях зазначається необхідність поєднання вимог щодо:

– енергоефективності;

– якості обслуговування (QoS), що зокрема включає вимоги до затримки, частоти помилок пакетів, тощо;

– надійності;

– безпеки.

При цьому, згідно з ITU-T L.1360 [70] енергоефективність стала одним із найважливіших аспектів як для поточної, так і для майбутніх телекомунікаційних інфраструктур, а у ITU-T L.1361 [71] описана методика визначення та метрики енергоефективності інфраструктури NFV. Для визначення енергоефективності

пропонується використання метрики «енергоефективність платформи NFVI з розгорнутою функцією VNF_j», оскільки віртуалізована функція сама по собі є програмним забезпеченням:

$$NFVI_EER_{VNF_j} = \frac{\left(Useful\ output\ of\ VNF_j\right)}{Power\ consumption_j} = \sum_{i=1}^{N}\left(\frac{U_{i,j}}{P_{i,j}} * w_i\right), \mathrm{j} \in \{1,2,\ldots\}, \quad (1.1)$$

де $U_{i,j}$ обсяг обробленого корисного навантаження VNF_j при рівні обслуговування i (під рівнем обслуговування мається на увазі ступінь масштабування віртуалізованої функції в залежності від поточного навантаження); $P_{i,j}$ споживана потужність платформи NFVI, на якій розгорнуто віртуалізовану функцію VNF_j при рівні обслуговування i.

Навантаження Network Slicing

Згідно з рекомендацією ITU-T Y.3156 [60] задачі логічного поділу мережі мають оброблятись з урахуванням таких вимог:

– Обробка повинна відбуватись так, щоб гарантувати виконання вимог QoS для послуг окремих рівнів мережі.

– Необхідно забезпечити постійний збір інформації щодо QoS на рівнях мережі. Результати аналізу якості обслуговування для логічних рівнів мережі мають надаватись у режимі реального часу.

– Дані повинні оброблятись з огляду на вимоги безпеки визначені у ITU-T Y.3111[72], b-ITU-T Y.2701[73].

Зазначимо, що тут маються на увазі вимоги щодо обслуговування саме навантаження поділу мережі (задач виокремлення логічних рівнів мережі, аналіз якості обслуговування на кожному з них, тощо), в той час як кожен логічний рівень (результат такого поділу) характеризується своїми вимогами щодо обслуговування, включаючи вимоги щодо енергоефективності, з огляду на специфіку сервісів, які надаються на цьому рівні. Так у специфікації ETSI TR 132 972 [74] зазначається, що визначення критеріїв енергоефективності для кожного рівня мережі ускладнено тим, що мережеві функції можуть спільно використовуватись на багатьох рівнях мережі та в свою чергу спільно використовувати фізичне обладнання, для якого

проводиться вимірювання енергоефективності. Окремо розглядається питання енергоефективності центрів обробки даних оператора інформаційно-комунікаційної мережі, які у тому числі забезпечують обслуговування навантаження Network Slicing. Згідно зі специфікацією, ключовими показниками енергоефективності центрів обробки даних ІКТ, є:

– енергоспоживання;

– продуктивність;

– повторне використання енергії;

– ступінь використання відновлювальної енергії.

Навантаження сервісів Edge Computing

У рамках концепції периферійних обчислень мається на увазі обслуговування цілої низки застосунків, що мають різні вимоги щодо обслуговування. Так, наприклад, згідно з рекомендацією ITU-T F.743.12 [63] щодо використання периферійних обчислень у системі відеоспостереження, зазначаються такі типові вимоги:

– аналіз відео та детектування позаштатних ситуацій у режимі реального часу;

– одночасне обслуговування великої кількості абонентських пристроїв (камер спостереження у контексті задачі відеоспостереження) без втрати QoE (англ. Quality of Experience – міра сприйняття клієнтом досвіду від користування послугою [80]);

– гарантована доступність системи не нижче рівня визначеного угодою про рівень надання послуг.

Аспект енергоефективності периферійних обчислень окремо розглядатиметься Міжнародною спілкою електрозв’язку у рекомендації L.EEMDC «Energy Efficiency in Micro Data Centre for Edge Computing», що знаходиться у розробці [75].

Навантаження bDDN

Щодо обробки навантаження bDDN згідно з рекомендацією ITU-T Y.3652 [65] окрім функціональних вимог висуваються також вимоги до продуктивності та безпеки:

– збір та аналіз даних про стан мережі має відбуватись у режимі реального часу;

– повинно бути забезпечене зберігання даних без втрат;

– час на аналіз даних обмежений зверху величиною максимально допустимого часу відповіді на запит, визначеного конкретним додатком;

– під час впровадження вимог пов’язаних bDDN слід застосовувати методи безпеки, такі як автентифікація, авторизація та контроль доступу, описані в ITU-T Y.2704 [76].

Оскільки аналіз Великих даних у мережі відбувається із використанням серверного обладнання центрів обробки даних, до обслуговування цього типу навантаження відносяться загальні вимоги щодо енергоефективності центрів обробки даних операторів ІКМ ITU-T L.1302 [77] та ETSI TR 132 972 [74], тобто забезпечення енергоефективності на основі показників енергоспоживання, продуктивності, повторного використання енергії, ступіня використання відновлювальної енергії.

Підсумуємо проаналізовані дані щодо типів обчислювального навантаження та їх вимог у таблиці (Таблиця 1.1).

Як бачимо, до ключових вимог належить забезпечення якості послуг QoS (англ. Quality of Service). До показників QoS у мережі 5G у відповідності до джерел [54, 62, 67] можна віднести:

– пікова швидкість передачі даних для висхідної/низхідної лінії зв'язку (Downlink/Uplink Peak Data Rate або $DL/UL\ PDR$);

– швидкість передачі даних для кінцевого користувача (Experienced Data Rate або EDR);

– затримка передачі даних “з кінця в кінець” (End-to-End latency, l_{E2E});

– частота помилок пакетів (Packet Error Rate, PER);

– ефективність використання частотного спектру (Spectral Efficiency, SE);

– обсяг мобільних даних для географічної зони (що фактично відображає якість покриття мережі);

– кількість підключених пристроїв (що фактично відображає ємність мережі);

– мобільність (здатність забезпечувати зв'язок для рухомих об'єктів при швидкості руху v).

Таблиця 1.1 Типи обчислювального навантаження сучасної ІКМ та вимоги до їх обслуговування згідно з рекомендаціями Міжнародної спілки електрозв'язку

Тип наванта-ження	**Приклад задач**	**Вимоги**	**Посилання на документи**
SDN та NFV	– маршрутизація, комутація, перетворення мережевих адрес; – оркестрація та реконфігурація мережевих ресурсів.	– забезпечення якості послуг QoS для кінцевого користувача; – енергоефективність; – надійність; – безпека.	ITU-T L.1360 [70] ITU-T L.1361 [71]
Network Slicing	– виокремлення логічних рівнів мережі; – аналіз вимог сервісів на кожному з логічних рівнів; – моніторинг виконання вимог щодо якості сервісу на кожному рівні.	– виконання вимог QoS для послуг окремих рівнів мережі; – аналіз якості обслуговування для логічних рівнів у режимі реального часу; – енергоефективність як сукупність показників: енергоспоживання, продуктивність, повторне використання енергії, ступінь використання відновлювальної енергії); – виконання вимог безпеки.	ITU-T Y.3156 [60] ETSI TR 132 972 [74] ITU T Y.3111 [72] b-ITU-T Y.2701 [73]
Edge Computing	– обробка даних системи відеоспостереження; – локальне кешування контенту.	– аналіз відео та детектування позаштатних ситуацій у режимі реального часу; – одночасне обслуговування великої кількості абонентських пристроїв без втрати QoE; – гарантована доступність системи не нижче рівня визначеного угодою про рівень надання послуг.	ITU-T F.743.12 [63] ITU-T F.743.10 [64]

Таблиця 1.1 (продовження)

Тип наванта-ження	Приклад задач	Вимоги	Посилання на документи
bDDN	– Аналіз даних про: – поточну конфігурацію мережі; – стан мережевого обладнання; – статистику мережевого трафіку; – історію помилок та збоїв у мережі; – кількість абонентів, що обслуговуються базовими станціями, тощо.	– збір та аналіз даних у режимі реального часу; – зберігання даних без втрат; – обмеження на максимально допустимий час відповіді на запит; – застосування методів безпеки; – енергоефективність.	ITU-T Y.3652 [65] ETSI TR 132 972 [74] ITU-T L.1302 [77]

При цьому, особливо актуальною є тема забезпечення якості надання послуг “з кінця в кінець” (тобто забезпечення так званої E2E QoS (End-to-end Quality of Service) [78, 79, 80]). В загальному випадку, оператор ІКМ може відповідати за якість надання сервісу лише в обмеженій частині мережі, яку він обслуговує. При цьому, у процесі надання сервісу можуть брати участь також ділянки мережі або обчислювальна інфраструктура, за яку не несе відповідальність безпосередньо надавач послуг (оператор мережі). E2E QoS має на меті гарантування якості обслуговування кінцевого користувача попри ці обставини.

Окрім QoS, як видно з таблиці (Таблиця 1.1), до вимог щодо обслуговування згаданих типів навантаження ІКМ належать:

– максимально допустимий час відповіді на запит;

– здатність роботи у режимі реального часу;

– доступність системи обслуговування навантаження;

– енергоефективність;

– надійність;

– безпека.

Виконання вимог щодо показників QoS, допустимого часу відповіді на запит та здатності працювати у режимі реального часу напряму залежить від

продуктивності обчислювальної інфраструктури, що здійснює обслуговування згаданих типів навантаження ІКМ. ***Доступність*** та ***енергоефективність*** обслуговування навантаження визначаються доступністю та енергоефективністю обчислювальної інфраструктури відповідно.

Таким чином, з метою підвищення ефективності процесу обслуговування наведених типів навантаження інформаційно-комунікаційної мережі, необхідно звернути увагу на вимоги щодо ***продуктивності***, ***енергоефективності*** та ***доступності системи*** обслуговування навантаження. Властивість та показники ***надійності*** є окремим аспектом обслуговування навантаження, що вивчається за допомогою методів теорії надійності [81]. ***Безпека*** обслуговування навантаження забезпечується впровадженням додаткових підходів щодо підвищення безпеки обробки даних. У цій роботі вимоги щодо надійності та безпеки обслуговування навантаження не розглядаються докладно та є питанням подальшого дослідження.

1.1.2 Характеристика процесу обслуговування навантаження

Об’єктом дисертаційного дослідження є процес обслуговування навантаження інформаційно-комунікаційної мережі. У цьому пункті наведено опис цього процесу, його учасників та показників ефективності.

Основними учасниками розподіленого обслуговування навантаження є обчислювальна інфраструктура у складі ІКМ представлена у вигляді ***розподіленої системи обробки даних*** та ***вхідне навантаження***.

1.1.2.1 Система розподіленого обслуговування навантаження у складі інформаційно-комунікаційної мережі та її параметри

Система розподіленої обробки обчислювального навантаження (або розподілена система) – це система просторово-розподілених обчислювальних компонентів, які:

– не розділяють спільної пам’яті;

– спілкуються за допомогою повідомлень;

– співпрацюють, вирішуючи єдину задачу [82].

Робота [83] «Розподілені системи. Принципи та парадигми» є фундаментальною працею у досліджуваній галузі. У цій роботі детально описані принципи, концепції та технології побудови розподілених обчислювальних систем. Розглянемо деякі з цих принципів, що безпосередньо стосуються проведеного дослідження.

Одиницею розподіленої системи є ***серверний кластер*** – певна кількість серверів, об'єднаних в групу, які утворюють єдиний ресурс (з єдиним або окремими сховищами даних). У контексті розподіленої системи, вузли якої розміщені географічно віддалено один від одного, доцільно розглядати серверний кластер, в якому кожен вузол оперує окремим сховищем даних. Недоліками систем з єдиним сховищем у цьому випадку є збільшення часової затримки при доступі до спільного сховища, а також високі витрати керування віддаленими даними. Варто відзначити, що під поняттям “серверний кластер” у роботі ми матимемо на увазі перш за все таку систему, в якій кожен вузол виконує корисну роботу (на відміну від систем з реплікацією даних або так званих FT-систем (Fault Tolerant) [84], метою яких є підвищення надійності системи за рахунок введення надлишковості ресурсів.

Фізичним елементом серверного кластера є ***сервер*** (обчислювальний вузол), а також мережеве обладнання, що з’єднує сервери високошвидкісними каналами зв’язку. Кластерні обчислення є вигідною альтернативою високопродуктивним суперкомп’ютерам [85] з економічної точки зору, оскільки вони, зазвичай, складаються зі стандартних обчислювальних вузлів (серверів), проте мають високу обчислювальну потужність завдяки розподілу обчислювальних задач між вузлами.

Крім фізичних елементів, обов’язковою складовою частиною розподіленої системи є ***програмне забезпечення розподіленої обробки даних*** (планувальники навантаження, метою яких є розподіл навантаження між обчислювальними вузлами; системи масштабування кластера, що здатні контролювати обсяг доступних обчислювальних ресурсів кластера в залежності від навантаження; ПЗ віртуалізації – програмне забезпечення, націлене на створення логічного рівня віртуальних ресурсів, абстрагованих від фізичної інфраструктури обчислювальної системи).

Серверні кластери та ряд допоміжних технічних засобів обробки, зберігання та передачі даних (наприклад, мережеве обладнання та системи зберігання даних) розміщуються на спеціальним чином спланованій території, що складається з зовнішніх майданчиків (анклавів), будівель, приміщень, зі встановленими інженерними системами забезпечення та обслуговуючим персоналом. Такий комплекс споруд та обладнання носить назву ***центр обробки даних (ЦОД).***

Між користувачем та оператором ЦОД укладається угода про рівень надання послуг (англ. SLA – Service Level Agreement), згідно з якою оператор ЦОД зобов'язується надавати послуги користувачеві із дотриманням вимог щодо визначених угодою параметрів. В загальному випадку угода SLA може містити вимоги щодо будь-яких параметрів, визначених домовленістю між оператором та користувачем, однак існують загальні рекомендації щодо складання SLA, що визначають ключові параметри (індикатори) якості обслуговування (англ. KPI – Key Performance Indicator) та їх допустимі значення. Розглянемо параметри розподіленої обчислювальної системи, визначені рекомендацією щодо SLA компанії Sun Microsystems [87]:

– $p_{ava\ \ SLA}$ – коефіцієнт готовності системи;

– $t_{max_{SLA}}$ – максимальний час відповіді на запит;

– $a_{SLA}|t = x$ – відсоток запитів із часом відповіді на запит не більше x секунд;

– Порядок реагування на неполадки (позаштатні ситуації):

- максимальний час відповіді на заявку користувача;
- максимальний час усунення неполадки;
- інтервал надання оновленої інформації про стан ситуації.

Коефіцієнт готовності системи згідно з SLA $\boldsymbol{p_{avail_{SLA}}}$– це визначене мінімальне значення коефіцієнта готовності, що визначається як відношення часу справної роботи до суми часу справної роботи і вимушених простоїв об'єкта (системи) з будь-якої причини, в тому числі і планових (до них відносяться профілактичні роботи, реконфігурації, зміна версій програмного забезпечення, модернізація обладнання, тощо), взятих за один і той же календарний термін [81].

Наприклад, коефіцієнт готовності 0,9999 означає, що протягом року постачальник має право не надавати сервіс споживачеві лише протягом 52 хв 35,7с, що становить 0,0001 від сумарного часу в 1 році [88]. Ймовірнісне визначення коефіцієнта готовності можна отримати переходячи від абсолютних значень часу роботи та простою до їх математичних сподівань. Коефіцієнт готовності характеризує ***доступність*** системи для користувача. Тобто, у вищезгаданому прикладі користувач отримає відповідь на свій запит, надісланий у випадковий момент часу t з імовірністю 0,9999. У ймовірнісному визначенні, доступність системи одночасно є ***ймовірністю успішної обробки запиту*** користувача.

Тоді, допустима згідно з SLA імовірність відхилення (втрати) запиту через будь-які причини визначається за допомогою рівності:

$$p_{loss_{SLA}} = 1 - p_{avail_{SLA}}. \tag{1.2}$$

Коефіцієнт готовності системи, а отже її доступність, залежить від багатьох факторів. Одна з причин недостатньої доступності – неправильний розрахунок кількості обчислювальних вузлів в системі, по відношенню до вхідного навантаження.

Параметри $t_{max_{SLA}}$ та $a_{SLA}|t = x$ опосередковано визначають необхідну ***продуктивність*** системи. Надаючи користувачеві гарантії щодо часу відповіді на запит, оператор розподіленої обчислювальної системи бере на себе зобов'язання забезпечити необхідний допустимий час відповіді кожної складової частини системи. Складність визначення цього параметру полягає у необхідності врахування можливого часу затримки у різних ланках системи "з кінця в кінець" (від користувача через усі ланки передачі та обробки назад до користувача) за формулою:

$$t_{max_{SLA}} = \sum_{i=1}^{N} t_{max_i}, \tag{1.3}$$

де t_{max_i} – максимальний час відповіді на запит для i-ої ланки системи, N – кількість ланок передачі та обробки запиту "з кінця в кінець", за умови, що не

всі ланки можуть знаходитись під безпосереднім контролем оператора системи (Рисунок 1.2).

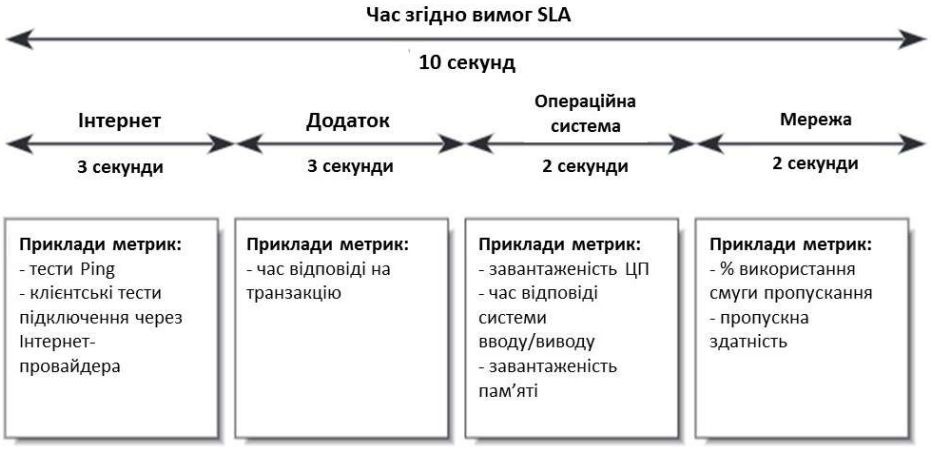


Рисунок 1.2 Параметр SLA "Максимальний час відповіді на запит" [87]

Параметр "Порядок реагування на неполадки" визначає правила дій персоналу ЦОД та визначає гарантований рівень сервісної підтримки для користувача. На цей параметр меншою мірою впливають технічні можливості системи.

Таким чином, типова угода SLA містить вимоги щодо ключових технічних параметрів розподіленої системи $p_{avail_{SLA}}$, $t_{max_{SLA}}$ та $a_{SLA}|t = x$. Допустимі значення цих параметрів повинні розраховуватись з урахуванням можливих втрат, затримок та поломок окремих ланок системи для гарантування так званої якості обслуговування користувача "з кінця в кінець".

Оскільки складовою частиною системи розподіленої обробки даних виступає серверний кластер, надалі розглянемо ці показники ефективності для серверного кластера та вважатимемо, що оптимізація вказаних показників для кожного кластера окремо впливає на покращення показників визначених в SLA в цілому.

Розглянемо розподілену обчислювальну систему, що складається з M серверних кластерів, кожен з яких в свою чергу складається з N обчислювальних вузлів. Кожен j-й вузол характеризується параметрами:

– V_{RAM_j}– об'єм оперативної пам'яті [ГБ];

– $V_{stora\quad j}$ – об'єм сховища даних [ГБ];

– C_j – продуктивність j-го вузла [од. навантаження / с] (проблему одиниць вимірювання обсягу навантаження та продуктивності відповідно докладніше розглянуто у підпунктах 1.1.2.2 та 4.3.1.1);

– $k_{core\,j}$ – кількість обчислювальних ядер вузла [шт.];

– $P_j = f_j(CPU_j)$ – модель енергоспоживання вузла – залежність потужності споживання j-го вузла від завантаженості його центрального процесора (ЦП).

Кожен з обчислювальних вузлів системи може знаходитись в одному з таких станів: «активний стан» – вузол обробляє певне навантаження та/або здатний прийняти на обробку нові обчислювальні роботи; «стан простою» – вузол є увімкненим, проте не обробляє жодної обчислювальної роботи в даний момент часу; «вимкнений» – вузол виведений з системи шляхом вимикання.

Для забезпечення функціонування серверного кластера в системі працює ***програмне забезпечення розподіленої обробки даних***, що відповідає за:

– злагоджену роботу всіх серверів;

– вирішення виникаючих в системі конфліктів;

– забезпечує формування і реконфігурацію серверного кластера у випадку збоїв, а також відновлення роботи додатків після збою за рахунок їх міграції;

– забезпечує розподіл навантаження між вузлами кластера;

– за необхідності здійснює масштабування кластера для адаптації кількості ресурсів під потреби обслуговування навантаження;

– здійснює моніторинг стану апаратного і програмного середовища.

Вхідне навантаження у загальному вигляді являє собою потік дискретних задач (обчислювальних робіт), що надходять до системи у певні (випадкові або детерміновані) моменти часу. Характеристика вхідного навантаження визначає алгоритми (підходи) обробки.

Одиницею вхідного навантаження є *обчислювальна робота (job).* Компонент роботи називається *задачею.* Фізично одна робота представляється у вигляді єдиного обчислювального *процесу* комп'ютера. *Підпроцеси або дочірні процеси* в свою чергу представляються задачами. У дисертаційній роботі оперуватимемо

терміном «робота» для визначення атомарного обчислювального завдання, яке може бути розміщене та виконане на одному з обчислювальних вузлів системи (тобто «робота» є одиницею розподілу навантаження).

В процесі розподілу навантаження кожна обчислювальна робота характеризується параметрами:

– $\Delta t_{max_i} = const[c]$ – максимальний час виконання роботи (або таймаут). Це період часу з моменту надходження роботи на обробку, по проходженню якого робота повинна бути завершена. Якщо робота не була успішно оброблена по проходженню часу Δt_{max_i}, вона виводиться із системи.

– Мінімально необхідний обсяг ресурсів для виконання роботи:

- мінімально необхідний об'єм оперативної пам'яті: $V_{RAM_{min\,i}}$[ГБ];
- мінімально необхідна кількість ядер процесора: $k_{cor\ \ min}$ [ядер] (у загальному випадку, для обчислювальної роботи може існувати можливість виокремлення її частини для паралельної обробки на різних ядрах процесора обчислювального вузла).
- мінімально необхідний об'єм постійного сховища даних: $V_{storage_{min\,i}}$[ГБ].

– *Пріоритет роботи* – необов'язковий числовий параметр, що може бути заданий для роботи та визначати пріоритет її виконання у порівнянні із іншими. Нехай пріоритет $priority$ визначається деяким дійсним невід'ємним числом (наприклад від 0,5 до 9, як у мережі LTE, де пріоритет може визначатись так званим QCI (QoS Class Identifier) – ідентифікатором класу QoS [89]. У мережах 5-го покоління цей параметр набув розвитку у так званому параметрі 5QI [90]). При цьому першочергової обробки вимагають задачі, для яких абсолютне значення пріоритету є меншим. Нехай за замовчуванням значення пріоритету становить найбільше з можливих, тобто задача не вважається пріоритетною, якщо не задано іншого.

Параметри та вимоги обчислювальної роботи визначаються її приналежністю до певного типу навантаження. Так, наприклад, обчислювальні роботи критичні до часу обслуговування (роботи щодо обробки даних системи відеоспостереження у

рамках Edge Computing, роботи віртуалізованих мережевих функцій щодо маршрутизації, комутації та перетворення мережевих адрес, тощо) матимуть більш жорсткі вимоги щодо максимального часу виконання Δt_{max_i}, але можуть бути менш затратними з точки зору вимог щодо мінімально необхідної кількості ядер процесора $k_{cores_{min\,i}}$ та обсягу оперативної пам'яті $V_{RAM_{min}}$ у порівнянні з обчислювальними роботами з аналізу Великих даних у рамках концепції bDDN.

Вхідне навантаження характеризується своєю інтенсивністю – обсягом навантаження, що надходить до системи в одиницю часу. Як правило, відомою є також добова статистика інтенсивності вхідного навантаження на систему.

У загальному вигляді процес обслуговування навантаження у розподіленій системі можна описати за допомогою схеми (Рисунок 1.3) [91, 92].

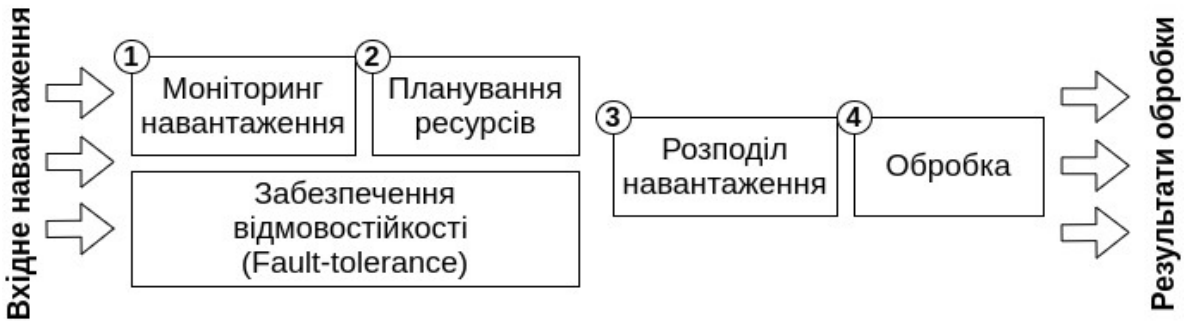


Рисунок 1.3 Процес обслуговування навантаження у розподіленій обчислювальній системі в загальному вигляді

Обчислювальні роботи надходять у систему у випадкові моменти часу τ.

В системі проводиться спостереження (***моніторинг***) інтенсивності надходження обчислювальних робіт (крок 1, Рисунок 1.3), на основі якого здійснюється планування ресурсів (крок 2, Рисунок 1.3). Під ***плануванням*** ресурсів мається на увазі обчислення та виділення необхідного обсягу ресурсів для обслуговування навантаження поточної інтенсивності. В процесі планування ресурсів, якщо наявний обсяг ресурсів не відповідає обсягу вхідного навантаження, використовуються механізми ***горизонтального*** або ***вертикального масштабування*** обчислювальної системи [93]. Під вертикальним масштабуванням мається на увазі нарощування обчислювальної потужності вузлів розподіленої

системи за рахунок заміни або додавання їх компонентів чи їх автоматичної активації або деактивації. Під горизонтальним масштабуванням мається на увазі збільшення або зменшення кількості обчислювальних вузлів в системі. Як вертикальне так і горизонтальне масштабування може бути здійснене динамічно в процесі роботи системи. Одночасно із процесами моніторингу та планування ресурсів забезпечується ***відмовостійкість*** системи. Під цим процесом мається на увазі підтримка певної надлишковості ресурсів у системі для її оперативної реконфігурації у випадку відмови основних ресурсів. Наступним кроком процесу обробки є ***розподіл*** (планування) навантаження (крок 3, Рисунок 1.3), тобто встановлення відповідності між обчислювальними роботами, що потребують обробки, та наявними ресурсами обробки у системі. Для планувальника навантаження розподілена система являє собою єдиний логічний набір ресурсів. Задачею планування навантаження у загальному випадку є розподіл обчислювальних робіт між фізичними вузлами обробки таким чином, щоб забезпечити обробку всіх робіт у найкоротший термін при оптимальному використанні обчислювальних ресурсів та з урахуванням пріоритетів обчислювальних робіт. За результатами кроку 3 обчислювальні роботи безпосередньо надходять на ***обробку*** займаючи виділені для них ресурси.

1.1.2.2 Показники та критерій ефективності процесу обслуговування навантаження

Як вже згадувалось, вимоги щодо обслуговування досліджуваних типів навантаження інформаційно-комунікаційної мережі включають вимоги щодо ***продуктивності***, ***енергоефективності*** та ***доступності*** системи. При цьому, загальними технічними показниками ефективності системи розподіленої обробки даних, що використовується в ІКМ для обслуговування навантаження, згідно з SLA є коефіцієнт готовності $p_{avail_{SLA}}$, максимальний час відповіді на запит $t_{max_{SLA}}$та відсоток запитів із часом відповіді на запит не більше x секунд $a_{SLA}|t = x$.

Визначимо взаємну відповідність цих показників ефективності та вимог щодо обслуговування навантаження ІКМ. Коефіцієнт готовності $p_{avail_{SLA}}$ характеризує

доступність системи, тому покращення цього показника прямо впливає на виконання вимог щодо доступності системи обслуговування досліджених типів навантаження ІКМ. Показники $t_{max_{SLA}}$ та $a_{SLA}|t = x$ описують властивість продуктивності системи. Енергоефективність розподіленої системи в цілому та серверного кластера зокрема є ключовим аспектом, що досліджується у роботі. Визначимо базові поняття, що можуть бути використані для оцінки енергоефективності:

– *Споживана потужність* – це робота, що виконується системою за одиницю часу (або енергія, передана за одиницю часу). Ця величина визначається за формулою:

$$P = \frac{A}{T} = \frac{W}{T} \text{ [Вт]}, \tag{1.4}$$

де A – сумарна робота, що виконується системою, W – спожита системою енергія, T – проміжок часу.

Потужність характеризує швидкість виконання роботи або швидкість перетворення енергії відповідно. Це скалярна величина, що вимірюється у Ваттах в системі СІ.

– *Обсяг спожитої енергії* – сумарний обсяг роботи, виконаний системою за період часу Т, що визначається формулою:

$$W = P * T \text{ [Вт} * \text{год]}, \tag{1.5}$$

де P – потужність, T – проміжок часу, що розглядається.

Різниця між споживаною потужністю та обсягом спожитої енергії є суттєвою в контексті енергоефективності, адже зниження споживаної потужності не завжди призводить до зниження сумарного обсягу спожитої енергії. Наприклад, при зниженні потужності процесора, він споживатиме менше електроенергії в одиницю часу, однак одна і та сама задача виконуватиметься на цьому процесорі значно повільніше, що в результаті призведе до незмінного обсягу спожитої енергії. Таким чином, важливим є факт врахування часу у формулі (1.5).

Крім того, для оцінки енергоефективності важливим є врахування корисної роботи з обробки даних, що була виконана при споживанні заданого обсягу електроенергії. Тому доречним є розгляд величини обсягу корисного навантаження, що обробляється системою.

Корисне навантаження – це цільове навантаження, при обробці якого ресурси системи використовуються безпосередньо за призначенням (на відміну від ресурсів, що витрачаються, наприклад, на роботу операційної системи, службові обчислення, узгодження у багатопроцесорній системі, тощо).

Обсяг корисного навантаження ω у загальному випадку визначається кількістю елементарних операцій із плаваючою комою, що повинні бути виконані системою, та вимірюється у позасистемних одиницях FLOP (від англ. Floating-Point OPerations – одиниця вимірювання обсягу навантаження, що відповідає кількості операцій з плаваючою комою, що повинні бути виконані обчислювальною системою для обслуговування навантаження [94]). Однак, вимірювання обсягу навантаження в одиницях FLOP на практиці може бути ускладненим, оскільки оцінка кількості операцій з плаваючою комою для довільної обчислювальної роботи є непростою задачею. Часто кількість операцій може бути невизначеною та залежати від вхідних даних, що обробляються алгоритмом у рамках обчислювальної роботи. Тому, у системах із невідомими заздалегідь обчислювальними роботами постає проблема визначення міри обсягу корисного навантаження. Однією з можливостей для вирішення цієї проблеми є введення відносної характеристики обсягу навантаження в тій чи іншій обчислювальній роботі на основі статистичних даних про час виконання робіт на однаковому АЗ. У дисертаційному дослідженні використовується така відносна характеристика обсягу навантаження для визначення продуктивності системи у рамках експериментальної перевірки ефективності запропонованого комплексного методу.

Отже, говорячи про ефективність процесу розподіленої обробки даних у серверному кластері, вважатимемо доцільним виділити такі показники:

1. ***Енергоефективність обробки даних*** $\boldsymbol{E_\Sigma}$ – обсяг корисного навантаження ω обробленого системою (серверним кластером) з N вузлів при споживанні обсягу енергії W. При обробці навантаження в N вузлах сумарний обсяг енергії розраховується як $W = \sum_{j=1}^{N} W_j$, а енергоефективність обробки даних відповідно визначається як:

$$E_\Sigma = \frac{\omega}{\sum_{j=1}^{N} W_j} \ [\frac{FLOP}{\text{Вт*год}}]. \tag{1.6}$$

Чим більший цей показник, тим більш енергоефективною є система, оскільки вона може обробити більшу кількість навантаження при меншому обсязі спожитої енергії.

2. ***Продуктивність обробки даних*** $\boldsymbol{C_\Sigma}$ (від англ. capability – продуктивність) – це обсяг навантаження, що обробляється системою за одиницю часу:

$$C_\Sigma = \frac{\omega}{T} \ [\frac{FLOP}{c}], \tag{1.7}$$

де ω – обсяг корисного навантаження, що обробляється системою; T – час обробки.

Показник продуктивності фактично визначає швидкість обслуговування навантаження обчислювальною системою.

3. ***Коефіцієнт готовності системи*** $\boldsymbol{p_{avail}}$ як показник її доступності.

Коефіцієнт готовності в імовірнісному визначенні (тобто у випадку визначення його через математичні сподівання часу роботи та простою), пов'язаний з імовірністю втрати запиту згідно з виразом:

$$p_{avail} = 1 - p_{loss}. \tag{1.8}$$

Якщо з певних причин (наприклад, через відсутність вільних ресурсів обробки даних) запит було втрачено, він потребує повторної обробки, що негативно впливає на QoE [80], тобто якість клієнтського досвіду.

Проаналізуємо взаємозв'язок між цими показниками на основі їх формальних визначень. Для прикладу розглянемо дві системи A і B з однаковою кількістю обчислювальних вузлів N. Обидві системи повинні обробити однаковий обсяг навантаження ω. Нехай система A є більш енергоефективною (тобто $E_{\Sigma_A} > E_{\Sigma_B}$). Згідно із формулою (1.6) це означає, що система A витратить на обробку навантаження ω менший обсяг енергії W. Обсяг спожитої енергії W

напряму пов'язаний із часом обробки, адже згідно з формулою (1.5) обсяг енергії визначається як $W = P * T$. Тоді можливими є два варіанта:

1. Система A витратила на обробку менше часу (тобто $T_A < T_B$). Тоді, згідно із формулою (1.7) продуктивність такої системи також буде більшою ($C_{\Sigma_A} > C_{\Sigma_B}$).

2. Система A витратила на обробку більшу або ж таку саму кількість часу (тобто $T_A \geq T_B$). У такому випадку менший обсяг спожитої системою A енергії пояснюється меншою потужністю споживання системи ($P_A < P_B$), що визначається потужністю споживання її компонентів (обчислювальних вузлів) та розподілом навантаження між цими компонентами. Тоді, згідно із формулою (1.7) продуктивність системи A буде меншою ($C_{\Sigma_A} < C_{\Sigma_B}$).

Тому, не можна стверджувати, що показник енергоефективності E_Σ та показник продуктивності C_Σ знаходяться у діалектичному протиріччі, і що збільшення одного показника обов'язково веде до зменшення іншого чи навпаки. Таким чином, бажаною є максимізація обох показників.

Для отримання формального виразу коефіцієнта готовності для досліджуваної системи p_{avail} необхідною є побудова математичної моделі розподіленої системи обслуговування навантаження (це питання докладно розглядається у підрозділі 2.2). Однак доступність системи, вочевидь, залежить від обсягу доступних у системі обчислювальних ресурсів (зокрема від кількості обчислювальних вузлів N) та обсягу вхідного навантаження ω, що потребує обробки. При однаковому обсязі навантаження ω краще значення коефіцієнта готовності матиме система із більшою кількістю вузлів N. При цьому, параметр N фігурує у формулі для показника енергоефективності (1.6) та впливає на кількість доданків у сумі $W = \sum_{j=1}^{N} W_j$. Природньо припустити, що збільшення кількості доданків веде до збільшення значення суми $\sum_{j=1}^{N} W_j$ та до зниження показника енергоефективності E_Σ відповідно. Однак, доданки W_j, що за своїм фізичним змістом являють собою обсяг спожитої енергії кожним вузлом системи окремо,

можуть різнитись для кожного j-го вузла та у свою чергу залежать від потужності споживання кожного вузла та часу, протягом якого вузол був задіяний в обробці ($W_j = P_j * T_j$). Таким чином, однозначно стверджувати, що довільна система з вищим коефіцієнтом готовності матиме нижчий показник енергоефективності також не є можливим. При цьому, важливо пам'ятати, що до мінімального значення коефіцієнта готовності висуваються вимоги у рамках угоди SLA: $p_{avail} \geq p_{avail_{SLA}}$.

Сформулюємо ***критерій ефективності*** для досліджуваної системи. У теорії прийняття рішень критерій ефективності – це правило, що формується на основі вибраних показників ефективності та дозволяє зробити цілеспрямований вибір з множини альтернатив [95]. При цьому виділяють критерії оптимальності (що використовується для прийняття оптимальних рішень згідно з концепцією максимізації корисності) та критерій придатності (що використовується для прийняття задовільних рішень згідно з концепцією обмеженої раціональності).

З огляду на вищесказане, критерій оптимальності для досліджуваного процесу обслуговування навантаження в інформаційно-комунікаційній мережі визначимо у вигляді:

$$K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$$

при $p_{avail} \geq p_{avail_{SLA}}$. (1.9)

Тобто у множині можливих конфігурацій системи обслуговування навантаження найбільш ефективною є така, що дозволяє отримати максимальне значення добутку показників енергоефективності E_{Σ} та продуктивності C_{Σ} при задовільному значенні коефіцієнта готовності системи p_{avail}. Задоволення критерію ефективності K_{opt} для кожного серверного кластера, що бере участь в обслуговуванні навантаження ІКМ (а саме, задач SDN та NVF, Network Slicing, Edge Computing та bDDN) позитивно впливає на продуктивність та енергоефективність інформаційно-комунікаційної мережі в цілому.

1.2 Існуючі підходи щодо підвищення енергоефективності процесу обслуговування навантаження в інформаційно-комунікаційній мережі

У цьому підрозділі проведено аналіз існуючих підходів щодо підвищення енергоефективності процесу обслуговування навантаження у розподілених обчислювальних системах у складі ІКМ.

Проблема енергоефективності розподілених обчислювальних систем активно досліджується у світі вже понад десятиріччя. Документ Міжнародної спілки електрозв'язку ITU-T L.1300 [96] поряд з рекомендаціями щодо побудови центрів обробки даних (включаючи вибір систем охолодження та енергопостачання) містить рекомендації щодо:

– вибору енергоефективного обчислювального та телекомунікаційного обладнання ЦОД;

– енергоефективного розгортання інформаційно-комунікаційних сервісів;

– енергоефективного керування телекомунікаційним обладнанням, сервісами та даними.

Згідно цих рекомендацій, загальними підходами щодо підвищення енергоефективності обслуговування навантаження ІКМ є:

– вибір енергоефективного обчислювального та телекомунікаційного обладнання та його регулярна атестація (аудит);

– розгортання сервісів з використанням підходів віртуалізації;

– розробка ефективного програмного забезпечення;

– виведення з експлуатації сервісів, що не використовуються та вимкнення обладнання у випадку його простою;

– консолідація поточних сервісів;

– постійний моніторинг енергоспоживання системи.

Тобто, згідно з [96] важливим є комплексний системний підхід, що враховує всі аспекти енергоефективності від характеристик АЗ до особливостей реалізації та розгортання програмного забезпечення.

Сучасна ідеалізована концепція енергоефективних обчислень була запропонована у 2008 році дослідниками Баррозу та Гольцле з компанії Google [3]. Ця концепція визначається як «енерго-пропорційні обчислення» і розглядає ідеалізовану модель обчислювального вузла, загальне споживання енергії якого пропорційне рівню його завантаженості. Автори дослідження також проаналізували реальну залежність енергоефективності обчислень від рівня завантаженості сервера (Рисунок 1.4).

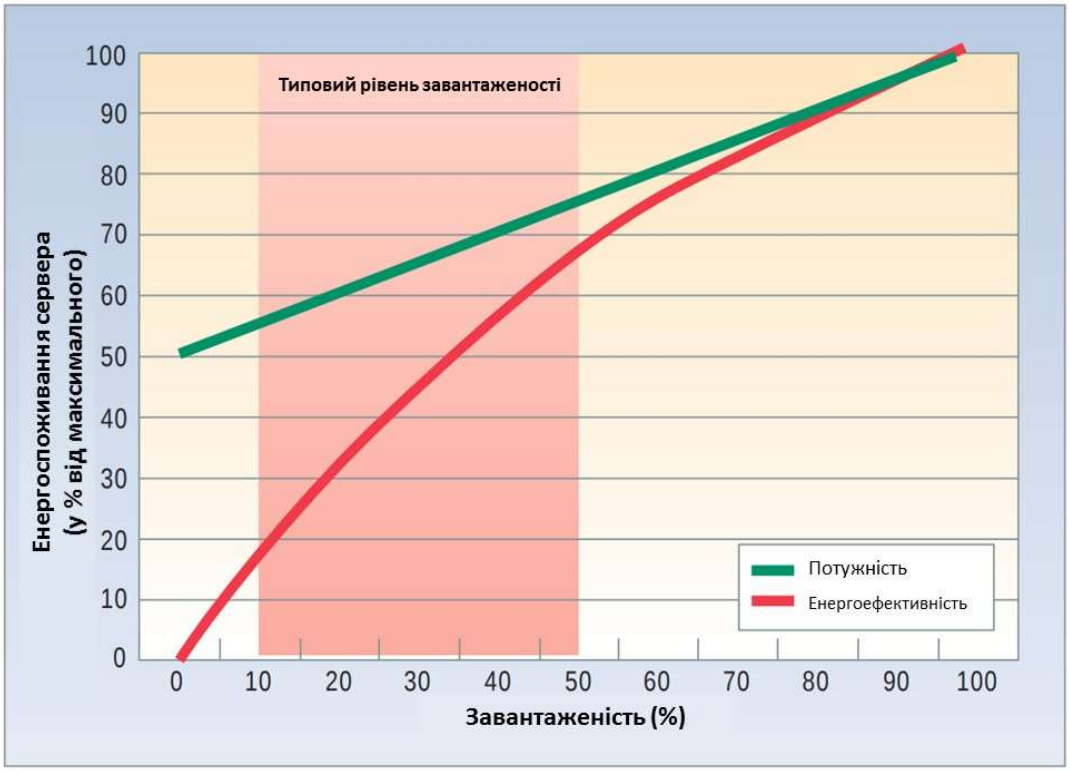


Рисунок 1.4 Залежність енергоефективності обчислень від завантаженості обчислювального вузла [3]

Як видно з рисунка (Рисунок 1.4), навіть коли завантаженість є нульовою (сервер не обробляє корисного навантаження), його енергоспоживання не є нульовим. Цей рівень енергоспоживання відповідає режиму простою. Зазвичай сервери працюють при завантаженості 10-50% (Рисунок 1.4), що і є причиною низької енергоефективності обчислень. Важливо відмітити, що чим більшою є завантаженість сервера, тим більша його енергоефективність. Тому, на думку авторів [3] основна ідея наближення до енергоефективних обчислень полягає в

забезпеченні максимального використання АЗ на піковій потужності роботи та повного уникнення простою обладнання.

Для наближення до моделі енерго-пропорційних обчислень розроблено багато підходів. Серед напрямів досліджень у цій галузі можна виділити групу підходів на рівні АЗ [3-15, 17], підходи консолідації та горизонтального масштабування [18 - 23], а також енергоефективне планування навантаження [25 - 33]. Проаналізуємо існуючі підходи за цими напрямами.

1.2.1 Підходи на рівні апаратного забезпечення

Автори роботи [4] запропонували просту класифікацію методів забезпечення енергоефективності обчислень в кластерних системах на рівні АЗ. Згідно цієї класифікації методи поділяються на статичні (такі, що не передбачають змін конфігурації системи під час її роботи) та динамічні (такі, що передбачають динамічну реконфігурацію системи в процесі обслуговування навантаження). Ідеєю статичних підходів є використання енергоефективних компонентів при побудові кластера, у той час як динамічні підходи мають на увазі адаптацію кластерної системи під час виконання відповідно до поточних потреб у ресурсах [4]. Динамічні підходи автори додатково розділили на такі, що базуються на енерго-масштабованих процесорах, енерго-масштабованій пам’яті та балансуванні (плануванні) навантаження.

Енерго-масштабованість компонентів системи означає їх здатність збільшувати або зменшувати енергоспоживання в залежності від рівня завантаженості. Енерго-масштабованість процесора забезпечується технологіями:

– динамічного масштабування напруги (dynamic voltage scaling або DVS) [4-8];

– динамічного масштабування частоти (dynamic frequency scaling або DFS) [9];

– комбінованого підходу щодо динамічного масштабування напруги та частоти (dynamic voltage and frequency scaling або DVFS) [10-14].

Прикладом технології енерго-масштабованої пам’яті можна вважати запропонований у роботі [15] підхід під назвою Memory Management Infra-Structure

for Energy Reduction (Memory MISER). Цей підхід забезпечує масштабування оперативної пам’яті в режимі офлайн або онлайн в залежності від потреб з боку навантаження та дозволяє досягти зменшення енергоспоживання елементів пам’яті на 70% [15] при деградації продуктивності роботи не більше ніж на 1%.

Підходи, що базуються на енергоефективному плануванні навантаження ми відносимо до окремого класу підходів у рамках нашого дослідження та розглянемо їх детальніше у пункті 1.2.3.

Автори роботи [16] запропонували розширену класифікацію енергоефективних підходів щодо підвищення енергоефективності обчислень (Рисунок 1.5). В цілому, автори об’єднують усі існуючі підходи щодо підвищення енергоефективності обчислень у 3 групи: підходи на рівні АЗ, підходи енергоефективного керування ресурсами та підходи щодо підвищення енергоефективності на рівні додатків. На рисунку (Рисунок 1.5) до підходів на рівні АЗ можна віднести усі названі класи, окрім підходів на рівні керування ресурсами, який відноситься скоріше до рівня програмного забезпечення і буде розглянутий далі.

Згідно із запропонованою класифікацією, статичні методи підвищення енергоефективності обробки даних додатково розділяють за рівнем застосування (на рівні ЦП (оптимізація інструкцій чи регістру) або всієї системи), у той час як динамічні підходи реалізуються або на рівні енерго-масштабованих компонентів, або на рівні керування ресурсами. Окрім енерго-масштабованих процесорів та пам’яті, автори також згадують енерго-масштабовані рішення щодо організації сховища даних. Це питання докладно проаналізовано у роботі [16].

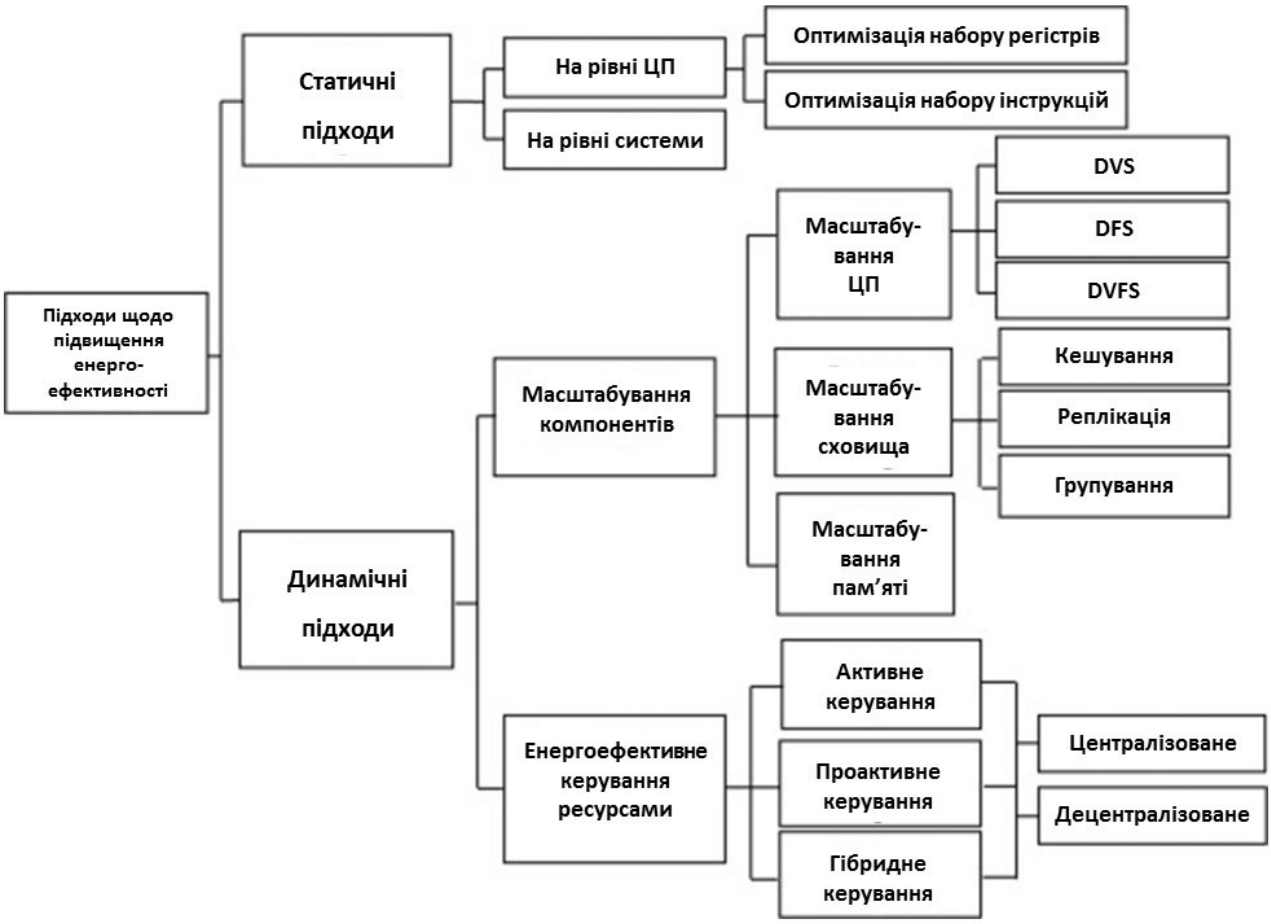


Рисунок 1.5 Класифікація підходів щодо підвищення енергоефективності обслуговування навантаження на рівні апаратного забезпечення [16]

Згідно дослідження [18], динамічні методи, за рахунок використання інформації про стан системи мають більшу ефективність, ніж статичні. Водночас, динамічні методи повинні додатково збирати, зберігати та обробляти інформацію про стан системи, внаслідок цього вони характеризуються високими накладними витратами у порівнянні зі статичними методами. Можливим також є комбінування статичних та динамічних методів, при якому для системи визначається ряд статичних шаблонів конфігурації, що базуються на статистичних даних, а за потреби ці шаблони коригуються із застосуванням динамічних підходів.

Статичні та динамічні підходи на рівні АЗ показують високу ефективність щодо економії електроенергії в системах розподілених обчислень, однак мають суттєвий недолік: впровадження цих підходів передбачає заміну апаратних компонентів обчислювальних вузлів, що вимагає додаткових витрат. Крім того, динамічне масштабування ресурсів системи може негативно вплинути на її

продуктивність (наприклад, у випадку стрибкоподібної зміни інтенсивності навантаження в системі). Однак, ми вважаємо, що використання підходів на рівні АЗ є особливо доцільним у поєднанні із підходами на рівні програмного забезпечення. Ці класи підходів можуть вдало доповнювати один одного у випадках роботи в гетерогенному обчислювальному середовищі, де апаратні рішення підтримуються лише частиною обчислювальних вузлів.

1.2.2 Горизонтальне масштабування та консолідація

Горизонтальне масштабування та консолідація належать до класу динамічних підходів керування ресурсами системи. Основною ідеєю цих підходів є зосередження процесу обслуговування навантаження на мінімальній кількості ресурсів (ущільнення або консолідація навантаження) з метою виведення частини ресурсів із активного стану (горизонтальне масштабування). Консолідація та масштабування зазвичай працюють разом і дозволяють знизити сумарне енергоспоживання системи за рахунок вимкнення частини обчислювальних вузлів або переведення їх у стан сну. Тобто ці підходи передбачають використовувати саме таку кількість серверів, яка на даний момент є необхідною для обробки поточного навантаження. Цей підхід зокрема підходить для віртуалізованих середовищ (тобто середовищ з абстрагованим від апаратної реалізації логічним об’єднанням ресурсів, представлених у вигляді так званих віртуальних машин (ВМ)), де ВМ можуть бути переміщенні між групами активних фізичних ресурсів.

Автори робіт [19, 93] запропонували підхід щодо динамічної консолідації ВМ та розробили програмне забезпечення OpenStack NEAT на його основі. Запропонований алгоритм консолідації ВМ вирішує 4 підзадачі:

1. Прийняття рішення про недостатню завантаженість вузла системи (з метою подальшого переміщення з нього усіх ВМ та переведення вузла у режим сну).

2. Прийняття рішення про надмірну завантаженість вузла з метою переносу частини ВМ з нього на інші вузли для уникнення погіршення продуктивності обчислень.

3. Вибір ВМ для переносу з перевантажених вузлів з усього набору ВМ на вузлі.

4. Розміщення ВМ, вибраних для переміщення, на інші активні або повторно активовані вузли.

Проведений авторами експеримент на тестовому наборі з 5 обчислювальних вузлів, показав, що динамічна консолідація ВМ здатна зменшити сумарне споживання енергії до 30% при обмеженому впливі на продуктивність обчислень. Для порівняння була використана система з рівномірним розподілом навантаження між вузлами, без врахування енергоспоживання вузлів та за умови їх постійного знаходження в активному стані. У роботі автори зазначили, що однією з важливих тем, які повинні бути розглянуті додатково, є врахування можливих динамічних змін умов якості обслуговування SLA, та вміння системи пристосовуватись до таких змін.

У віртуалізованих середовищах нагальною проблемою є також ресурсозатратність самого процесу міграції ВМ, що у найгіршому випадку може нівелювати виграш від горизонтального масштабування щодо енергоефективності. У роботі [20] досліджується проблема ресурсозатратності живої міграції ВМ (динамічної міграції без відключення), що здійснюється з метою горизонтального масштабування серверного кластера. В залежності від часу, що потрібен для міграції, цей процес може потребувати той чи інший обсяг обчислювальних ресурсів та електроенергії, а також спричинити нестачу обчислювальних ресурсів через тимчасову недієздатність ВМ. Автори роботи пропонують математичні моделі для оцінки часу, необхідного для живої міграції ВМ, що являють собою лінійні комбінації параметрів використання ресурсів (зокрема загальної кількості вилучених інструкцій ЦП, загальної кількості промахів рядків L3-кешу ЦП та кількості «брудних» сторінок, які спостерігаються на вихідному сервері під час міграції). Завдяки використанню методів простої і множинної лінійної регресій автори якісно та кількісно дослідили значення цих параметрів та їх вплив на час міграції.

Автори роботи [21] запропонували створювати розклад відключення вузлів серверного кластера (горизонтального масштабування) за допомогою використання алгоритмів кластеризації середніх зв'язків і K-середніх. За допомогою цих алгоритмів автори класифікують рівні завантаженості серверного кластера як «низький», «середній» та «високий» (на основі статистичних даних про навантаження) і складають розклад відключення серверів, що включає моменти та тривалість відключення, згідно з визначеними рівнями завантаження. Серед недоліків такого підходу можна виділити неспроможність системи (без введення додаткових механізмів) реагувати на динамічні зміни навантаження, а також той факт, що при відключенні не беруться до уваги індивідуальні характеристики обладнання.

Автори роботи [22] описують підхід Autopilot, що використовується компанією Google для керування ресурсами масштабних серверних кластерів у компанії, а також розроблене на його основі програмне забезпечення. Основною проблемою, на вирішення якої націлена робота, є зменшення так званого «зазору» (англ. "slack") між максимально можливим та реальним використанням наявного ресурсу. Autopilot базується на ідеї поєднання горизонтального та вертикального масштабування, однак, автори підходять до вирішення проблеми із точки зору обчислювальних робіт, а не обладнання: завдяки використанню планувальника навантаження Borg у досліджуваній системі існує можливість коригувати вимоги обчислювальних робіт щодо мінімально необхідних ресурсів. Тобто, наприклад, якщо було визначено, що обчислювальна робота вимагає більше ресурсів, ніж мінімально необхідно насправді, наступний раз вимоги цієї роботи будуть скориговані (автори називають такий підхід вертикальним масштабуванням). Крім того, існує можливість змінювати кількість задач у рамках обчислювальної роботи, що називається горизонтальним масштабуванням. Для реалізації цих підходів автори використовують алгоритми машинного навчання та евристичні алгоритми. Цей підхід дозволяє якомога точніше ставити у відповідність обчислювальні роботи та обсяг ресурсів для їх обробки, що дозволяє знизити рівень простою обладнання за рахунок уникнення непотрібної резервації ресурсів. Однак, такий

підхід є перш за все підходящим для застосування у середовищах із постійним, передбачуваним навантаженням, для якого є змога зібрати історичні дані про виконання кожного типу обчислювальних робіт. У системі із більш різноманітним та непередбачуваним навантаженням (як, наприклад, задачі від користувацьких додатків, що надходять на обробку до периферійної хмари), такий підхід може виявитись менш ефективним.

У роботі [23] проаналізовано використання ряду політик горизонтального масштабування мультисерверної системи, що носять назву NEVEROFF, INSTANTOFF та SLEEP. Ці політики передбачають переведення частини серверів у режим сну при різних розподілах навантаження у серверному кластері. Автори висувають гіпотезу, що для мультисерверної системи з пуассонівським вхідним потоком запитів політик NEVEROFF, INSTANTOFF і SLEEP достатньо для отримання оптимальної політики горизонтального масштабування. В роботі [23] докладно описано кожну з цих політик та запропоновано підходи до вибору однієї з них. Крім того, в роботі описано та запропоновано нову політику DELAYEDOFF, згідно з якою сервери вмикаються при приході нового завдання, але вимикаються лише по проходженню певного періоду часу після обробки завдання.

Питання, які потребують вирішення при використанні підходів консолідації та горизонтального масштабування: як прогнозувати зміни робочого навантаження, щоб вчасно масштабувати систему (аби уникнути погіршення продуктивності)? Які ресурси з усіх наявних бажано використовувати більш інтенсивно? Який обсяг ресурсів є необхідним для обслуговування навантаження у кожен момент часу та як часто доцільно проводити масштабування?

1.2.3 Енергоефективний розподіл навантаження

Згідно з класифікацією наведеною у [24] окрім консолідації та горизонтального масштабування важливим класом підходів щодо підвищення енергоефективності розподіленої обробки даних є енергоефективне планування (англ. scheduling).

Основною задачею процесу розподілу навантаження є встановлення відповідності між наявними ресурсами обробки даних та навантаженням, що потребує обробки. Найпростішими підходами щодо розподілу навантаження є FCFS (англ. First Come First Served) та RoundRobin (RR) [25]. Алгоритм FCFS розподіляє задачі згідно їх порядку у черзі. Якщо ресурсів для виконання наступної задачі виявляється недостатньо, алгоритм чекає їх звільнення. RR в свою чергу визначає порядок зайняття обчислювальних ресурсів. Згідно з RR ресурси "заповнюються" задачами один за одним по колу. Ці підходи не враховують показник енергоефективності у процесі розподілу задач.

Як видно з рисунка (Рисунок 1.4), чим більшою є завантаженість сервера, тим більшою є його енергоефективність. Тому, метою енергоефективного розподілу задач є встановлення відповідності між наявними обчислювальними ресурсами та навантаженням, таким чином, щоб звести до мінімуму простій обладнання та підвищити ступінь завантаженості доступних ресурсів.

Енергоефективний алгоритм планування навантаження CTES – Cooperative Two-Tier Energy-Aware Scheduling запропоновано та описано у роботі [26]. Автори розглядають дворівневий підхід до розподілу задач, що мають бути оброблені у режимі реального часу. На першому, локальному рівні пропонується розподіляти ресурси одного сервера між задачами так, щоб кожна задача отримала мінімальну обчислювальну потужність, яка необхідна для її обробки. Ресурси, що залишилися після такого розподілу, розподіляються між всіма задачами. На другому, глобальному рівні, задачі розподіляються між серверами кластера згідно з однією із описаних у роботі технік: Predictive Energy, Predictive Total Time, ET2. Ці техніки одночасно враховують параметр енергоефективності та продуктивності при розподілі задач. Недоліком даного алгоритму є те, що автори вважали модель енергоспоживання сервера (функцію залежності потужності споживання від навантаження центрального обчислювального вузла) лінійною. Насправді така модель не точно відображає характер залежності споживаної потужності від завантаженості сервера. Натомість у даному дисертаційному дослідженні пропонується проведення попередньої атестації кожного сервера кластеру для

отримання реальної залежності потужності споживання від завантаженості для кожного обчислювального вузла.

У роботі [27] описано стратегію розподілу задач Min_c. Дана стратегія враховує різноманітність задач, що приходять на обробку та відповідну різноманітність ресурсів, яких вони потребують. Недоліком описаного у статті [27] алгоритму є те, що енергетична модель, яку використали автори, хоч і має нелінійний характер, близький до дійсності, проте використана модель має єдиний вигляд для всіх машин в незалежності від їхніх характеристик. Натомість у даній дисертаційній роботі пропонується індивідуально визначати модель енергоспоживання для кожної машини окремо.

У роботі [28] описано підхід до енергоефективного розподілу задач, що має назву «Алгоритм P». Даний алгоритм автори пропонують застосовувати при розподілі задач між кількома незалежними серверними кластерами. Згідно запропонованого авторами підходу, вхідним параметром для «Алгоритму P» є значення H_{diff}, що визначає допустиму різницю між мінімальним та максимальним часом очікування задачі у черзі до кластера, за якого алгоритм буде давати економію електроенергії. Алгоритм вибиратиме кластер, у якому задача буде оброблена із найменшими затратами енергії. Для кожного кластера планувальник навантаження C_j оцінює вартість електроенергії, що буде затрачена на обробку задачі, і параметр h_j – відношення загальної площі задач до ширини кластера (цей параметр визначає оцінку часу перебування задачі у черзі). Якщо $\max(h_j) - \min(h_j) > H_{diff}$, то задача розподіляється на обробку на кластер з мінімальним значенням h_j. Інакше, задача розподіляється на кластер із мінімальною вартістю енергії. Автори роботи [28] розглядали у якості оцінюваних параметрів час очікування задачі у черзі та вартість електроенергії, вважаючи її різною для різних кластерів. У випадку застосування підходу у межах одного серверного кластера, оцінюваними параметрами можуть виступати час знаходження задачі у черзі до кожного із серверів та вартість обробки задачі на кожному окремому сервері. Серед недоліків вищеописаного підходу можна

виділити те, що автори не оцінюють параметр продуктивності кожного окремого сервера. Вибір вузла обробки здійснюється на основі поточного стану кластера та завантаженості його вузлів. Такий підхід може негативно вплинути на продуктивність обчислювальної системи в цілому.

Однією з найбільш розповсюджених стратегій енергоефективного планування задач є стратегія Backfill [29]. Базовий варіант цієї стратегії був запропонований компанією IBM та став підґрунтям для розвитку численних модифікацій та покращень цього підходу [30, 31, 32].

Backfill – це стратегія оптимізації розподілу навантаження, яка дозволяє планувальнику краще використовувати наявні апаратні ресурси, розподіляючи обчислювальні задачі не строго в порядку їх надходження у чергу, а у довільному порядку в залежності від розміру задачі. Основною ідеєю стратегії Backfill є “щільне заповнення” вільних ресурсів, довільними задачами із черги, для виконання яких обсяг цих ресурсів є достатнім. Стратегія Backfill передбачає, що кожна обчислювальна робота характеризується максимальним часом виконання, після проходження якого робота автоматично знімається з обробки. Спершу планувальник резервує ресурси для “великих” обчислювальних робіт у черзі. Він розраховує час початку виконання роботи з урахуванням моменту звільнення необхідних ресурсів у системі. Після чого планувальник оцінює, чи можна використати ресурси, що залишаються не задіяними до початку виконання “великої” роботи для виконання “швидких” робіт із черги до того моменту, коли “велика” робота повинна буде надійти в обробку. Таким чином, планувальник підбирає із черги задачі з меншим пріоритетом, але меншого обсягу, заповнюючи «пусті місця» на вузлах обробки. На рисунку (Рисунок 1.6) наведено приклад застосування стратегії Backfill та її порівняння з підходом FCFS, за показниками середньої завантаженості процесора вузла та сумарного часу обробки задач [30]. Як видно з рисунка, стратегія Backfill дозволяє скоротити час виконання обчислювальних робіт при сталому обсязі ресурсів за рахунок більш щільного їх заповнення.

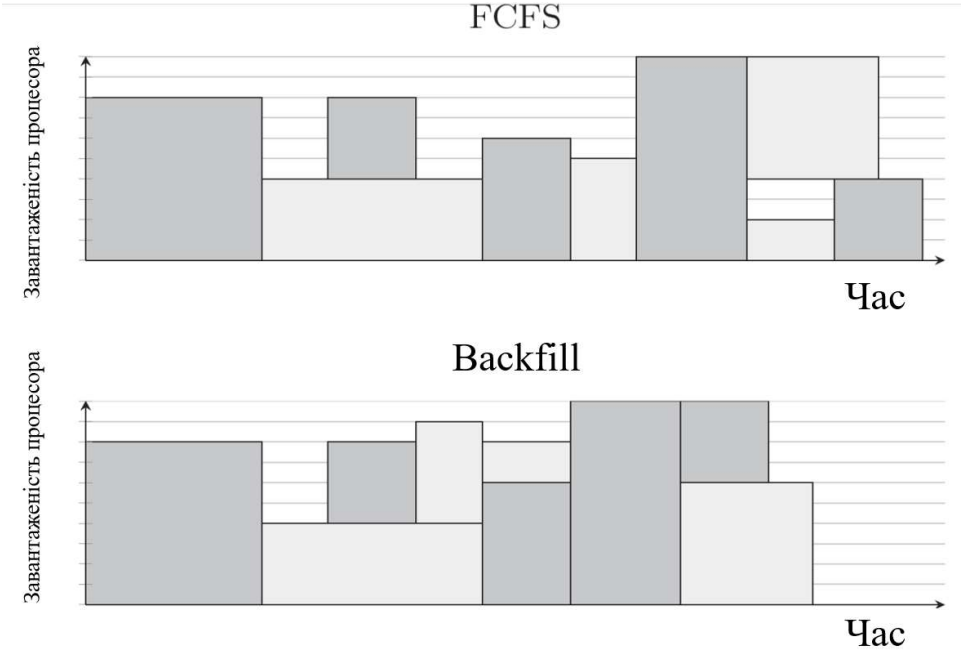


Рисунок 1.6 Принцип роботи стратегії Backfill у порівнянні з FCFS [30]

Стратегія Backfill та її модифікації дозволяють досягти підвищення енергоефективності за рахунок підтримки високого рівня завантаженості серверів. Однак, ефективність виключно цього підходу без використання додаткових технік не перевищує у значній мірі ефективність найпростіших підходів FCFS та RR. Це пояснюється тим, що всі ресурси системи підтримуються в активному стані та споживають електроенергію, навіть коли у них немає потреби (так само, як це відбувається і при використанні FCFS та RR).

Іншим відомим підходом щодо енергоефективного розподілу навантаження є запропонований у роботі [33] алгоритм The Most-Efficient-Server First (MES-first). Згідно даного підходу, роботи із черги розподіляються спершу на найбільш енергоефективні сервери, згодом на менш енергоефективні, згідно позиції сервера у сортованому списку. Розподіл задач припиняється, якщо не лишається задач у черзі на обробку, або якщо черги до всіх серверів заповнені. Цей підхід виявляє найбільшу ефективність у поєднанні із підходами масштабування ресурсів, коли найменш енергоефективні сервери підтримуються у режимі сну поки ресурсів активних енергоефективних серверів вистачає для обробки поточного навантаження. Недоліком запропонованого у роботі [33] підходу є те, що на відміну

від стратегії Backfill він не має на меті підтримку максимального рівня завантаженості кожного окремого сервера. Як видно з рисунка (Рисунок 1.4), недостатня завантаженість вузла обробки призводить до зниження енергоефективності його роботи.

Таким чином, алгоритм енергоефективного планування навантаження Backfill є більш прийнятним для вирішення задачі підвищення енергоефективності обробки даних з огляду на концепцію енерго-пропорційних обчислень та може бути використаний у рамках дисертаційного дослідження. Однак, виникає задача удосконалення Backfill з метою покращення його роботи за умов недостатньої завантаженості системи, а також пошуку балансу між енергоефективністю та продуктивністю обробки даних за умови помірної завантаженості системи.

ВИСНОВКИ

1. Аналіз специфікацій та рекомендацій Міжнародної спілки електрозв'язку та Європейського інституту телекомунікаційних стандартів показав, що основною особливістю сучасної ІКМ та зокрема стільникової мережі 5G є її програмованість, що забезпечується технологіями та концепціями SDN, NFV, Network Slicing, Edge Computing, bDDN. Задачі, що вирішуються у рамках цих концепцій формують обчислювальне навантаження, що потребує обслуговування за допомогою розподіленої інфраструктури обчислень у складі ІКМ.

2. Аналіз вимог щодо обслуговування визначених типів навантаження ІКМ дозволив виділити основні показники ефективності процесу обслуговування у серверному кластері як одиниці розподіленої інфраструктури. Цими показниками є енергоефективність E_{Σ} та продуктивність C_{Σ} обслуговування обчислювального навантаження, а також коефіцієнт готовності системи p_{avail} як показник її доступності. На основі цих показників сформульовано критерій ефективності системи K_{opt}.

3. Проведено огляд існуючих методів підвищення енергоефективності процесу обслуговування навантаження в ІКМ та проаналізовано їх ефективність щодо визначених показників (енергоефективності, продуктивності та доступності системи). Серед існуючих методів виділяють підходи на рівні апаратного та програмного забезпечення, які в свою чергу поділяють на класи підходів щодо горизонтального масштабування та консолідації, а також енергоефективного розподілу навантаження.

4. Основним недоліком підходів на рівні апаратного забезпечення є складність та високі витрати на їх впровадження.

5. Підходи масштабування та консолідації дозволяють суттєво знизити сумарне енергоспоживання обчислювальної системи, однак можуть негативно впливати на продуктивність та доступність системи, особливо у випадку динамічних змін інтенсивності навантаження, що є критичним з огляду на необхідність гарантувати виконання вимог до якості надання послуг для

навантаження ІКМ. Крім того, існуючі підходи щодо масштабування та консолідації не враховують індивідуальні характеристики енергоспоживання обчислювальних вузлів, що призводить до неоптимального використання обчислювальних ресурсів.

6. Серед підходів енергоефективного планування навантаження було окремо відзначено алгоритм планування навантаження Backfill, основною перевагою якого є мінімізація простою обчислювальних вузлів за рахунок щільного розподілу обчислювальних робіт. Однак ефективність цього підходу зменшується у випадку невисокої інтенсивності вхідного навантаження, оскільки він не забезпечує відсутність простою в умовах збитковості обчислювальних ресурсів.

7. Окремим виявленим недоліком існуючих методів є те, що жоден із них не вирішує задачу підвищення ефективності обслуговування навантаження з урахуванням усіх трьох виділених показників ефективності (енергоефективності, продуктивності та доступності системи), що є ключовими для обслуговування навантаження ІКМ.

8. Визначено необхідність систематизації та формалізації процесу обслуговування навантаження в ІКМ з урахуванням показників енергоефективності, продуктивності та доступності системи.

РОЗДІЛ 2
МЕТОДИ ДОСЛІДЖЕННЯ ПРОБЛЕМИ ЕНЕРГОЕФЕКТИВНОСТІ ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ

Обслуговування навантаження в інформаційно-комунікаційній мережі є складним процесом, на який впливають такі фактори як характеристики та обсяг обчислювального навантаження, кількість та характеристики вузлів обробки, програмне забезпечення, правила та політики розподілу навантаження, тощо. При цьому, для оцінки цього процесу існує ряд показників та вимог щодо їх значень. Тому, виникає необхідність встановлення взаємозв'язків між впливаючими факторами та показниками ефективності досліджуваного процесу. Для цього необхідно побудувати модель процесу, що дозволила б якісно аналізувати впливаючі фактори у сукупності. Серед існуючих методів формалізації складних систем було обрано онтологічну модель, яка дозволяє відображати складні взаємозв'язки у рамках досліджуваного процесу [34].

Окрім систематизації взаємозв'язків між впливаючими факторами та показниками ефективності системи, існує необхідність математичного моделювання процесу обслуговування навантаження в інформаційно-комунікаційній мережі. Це дозволить охарактеризувати ці взаємозв'язки не тільки якісно, але й кількісно. Для цього у розділі описано математичне моделювання досліджуваного процесу як системи масового обслуговування (СМО) [35].

Окремо розглянуто також проблему моделювання енергоспоживання обчислювальних вузлів розподілених ЦОД у складі інформаційно-комунікаційної системи.

Таким чином, цей розділ присвячений методам дослідження проблеми енергоефективності обслуговування навантаження інформаційно-комунікаційної мережі, а саме: методу формалізації складних процесів за допомогою онтологічної моделі, методам теорії масового обслуговування, математичним методам функціонального аналізу (апроксимації, інтерполяції).

2.1 Формалізація процесу обслуговування навантаження в інформаційно-комунікаційній мережі

Для оператора ІКМ все більшої актуальності набуває проблема побудови та підтримки системи обробки даних з можливістю гнучкого керування якістю послуг та масштабування. Такі системи обробки даних мають комплексну внутрішню структуру з багатьма взаємопов'язаними параметрами, що ускладнює їх аналіз, керування та розширення. Для спрощення подальшої роботи пропонується представити досліджувану систему у формалізованому вигляді, виділивши всі її сутності, параметри та взаємозв'язки.

2.1.1 Аналіз існуючих підходів щодо формалізації складних систем та процесів

Існує багато підходів до формалізації складних систем. Зокрема, система може бути представлена за допомогою реляційної моделі [97], об'єктно-орієнтованої моделі [98, 99], мережевої моделі [100], онтологічної моделі [101, 102].

Реляційна модель зосереджена на упорядкуванні даних у двовимірних таблицях. Перевага реляційної моделі даних полягає у простоті та зручності фізичної реалізації на ЕОМ (електронно-обчислювальна машина). Основними недоліками реляційної моделі є відсутність стандартних засобів ідентифікації окремих записів та складність опису ієрархічних та мережевих взаємозв'язків між сутностями системи.

Об'єктно-орієнтована модель представлення даних оперує такими поняттями, як клас і об'єкт. Класи визначають структуру даних і набір атрибутів (текстовий рядок, ціле число, зображення тощо). Екземпляри класу (об'єкти) зберігають внутрішню будову описану на рівні класу і можуть утворювати довільні ієрархічні структури. Як правило, системи, засновані на об'єктно-орієнтованій моделі даних, є функціональними, гнучкими, але в той же час більш складними. Будучи ієрархічними, ці системи мають обмеження: варіант використання, коли дочірній клас має більше одного батьківського класу, тут не підтримується. Для вирішення

цієї проблеми існує мережева модель. У цій моделі класи та відносини представлені у вигляді графа, що робить цю модель набагато гнучкішою.

Онтологія – це спроба всебічної та детальної формалізації певної галузі знань за допомогою концептуальної схеми [102]. За допомогою онтологічної моделі дані можуть бути представлені у вигляді сукупності різних типів інформаційних об’єктів та зв’язків між ними. Головною перевагою онтологічної моделі є те, що на відміну від об'єктно-орієнтованої та мережевої моделі, вона дозволяє детально описувати не лише класи та об'єкти, а й складні семантичні зв'язки між ними. Ці переваги дозволили обрати онтологічну модель для формалізації системи розподіленої обробки даних в ІКМ.

2.1.2 Онтологічна модель процесу обслуговування навантаження

Онтологічна модель дозволяє структурувати та систематизувати дані системи обробки інформації та прозоро відобразити взаємозв’язки між параметрами системи для спрощення її аналізу та масштабування.

Будучи структурованим поданням інформації в якійсь предметній області, кожна онтологічна модель базується на необроблених вихідних даних, що зберігаються в певній інформаційній базі даних (Рисунок 2.1). Прикладами таких вихідних даних є записи абонентів у системі виставлення рахунків, статистика щоденного навантаження в системі обробки даних, номінальні параметри обладнання для обробки даних тощо. Онтологічна модель перетворює ці необроблені дані у ***знання***. Моделюючи систему обробки даних як онтологію, ми можемо виділити 3 окремі структурні частини системи. Ці частини відповідають трьом підкомпонентам онтологічної моделі відповідно:

1. Онтологія компонентів системи розподіленої обробки обчислювального навантаження (сервери, ПЗ для організації обробки даних, тощо);
2. Онтологія вхідних параметрів (вхідні джерела та параметри робочого навантаження, тощо);
3. Онтологія показників оцінки якості системи (продуктивність, енергоефективність, тощо).

Рисунок 2.1 Загальна структура запропонованої онтологічної моделі системи розподіленої обробки даних

Розглянемо запропоновані підсистеми окремо. Обладнання та програмне забезпечення для організації обробки обчислювального навантаження представляють ***підсистему компонентів*** обчислювальної системи. Онтологія цих компонентів відображає фізичну структуру самої системи. Система обробки обчислювального навантаження складається з серверних кластерів, що в свою чергу складаються з N фізичних вузлів (серверів). Кожен сервер характеризується обсягом обчислювальних ресурсів (кількістю обчислювальних ядер, обсягом оперативної пам'яті, тощо) і працює зі спеціалізованим програмним забезпеченням для організації розподіленої обробки даних, призначеним для керування ресурсами системи обробки, розподілу обчислювальних завдань, забезпечення відмовостійкості та надійності системи, тощо.

Онтологія вхідних параметрів охоплює опис робочого навантаження, його параметрів та частково – джерел. У розглянутій системі розподіленої обробки оператора ІКМ джерелами робочого навантаження є зокрема Великі дані оператора (мережева статистика, дані про активність абонентів, тощо), віртуалізовані мережеві функції, дані додатків, що потребують попередньої обробки в периферійній хмарі, тощо. Обчислювальна робота – це одиниця робочого

навантаження. Кожна обчислювальна робота характеризується своїми вимогами до ресурсів системи та часу обробки, які залежать від типу джерела навантаження.

Онтологія критеріїв оцінки якості системи або ***онтологія вихідних параметрів*** охоплює опис тих критеріїв, які оцінюються для системи та показують ефективність її функціонування. Для ефективної обробки обчислювального навантаження оператору ІКМ потрібно забезпечити:

– Виконання вимог щодо коефіцієнта готовності системи $p_{avail} \geq p_{avail_{SLA}}$ залежно від типу навантаження (наприклад, коефіцієнт готовності для внутрішніх завдань bDDN щодо обробки Великих даних мережевої статистики може бути нижчим, ніж коефіцієнт готовності периферійної хмари, оскільки обробка даних у периферійній хмарі передбачає пряму взаємодію з абонентом, часто у режимі реального часу).

– Достатню продуктивність системи обробки даних C_{Σ} (під «продуктивністю» тут мається на увазі кількість оброблених обчислювальних робіт за одиницю часу. Тобто, чим більша продуктивність системи обробки даних, тим вища пропускна здатність обробки).

– Високу енергоефективність системи обробки обчислювального навантаження E_{Σ}.

Ці параметри описуються онтологією критеріїв оцінки якості системи.

На рисунку (Рисунок 2.2) зображено запропоновану онтологічну модель системи, що формально описується за допомогою виразу:

$$O = \{C, A, R, T, F, D\}, \tag{2.1}$$

де C – множина класів; A – множина атрибутів; R– множина відношень; T – множина типів значень; F– множина обмежень на значення відношень; D – множина екземплярів класів.

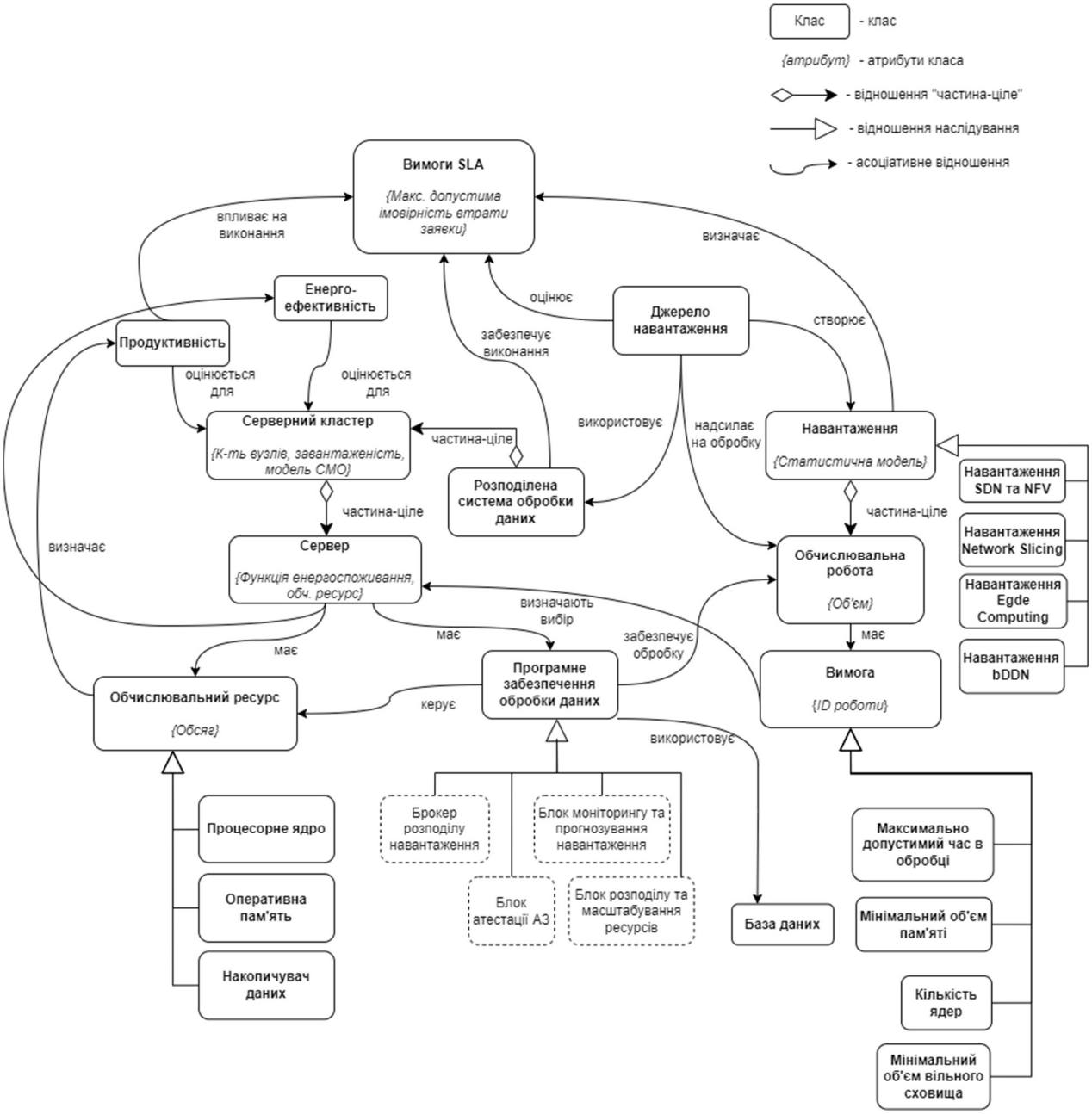


Рисунок 2.2 Запропонована онтологічна модель системи розподіленого обслуговування навантаження інформаційно-комунікаційної мережі

Множина класів онтології: $C = \{C_1, C_2, \ldots, C_{25}\}$

C_1 – *Розподілена система обробки даних.* Цей клас описує поняття системи розподіленої обробки даних як фізичну сутність.

C_2 – *Джерело навантаження.* До цього класу відносяться поняття, пов’язані з суб’єктами, що прямо або опосередковано використовують розподілену обчислювальну систему (користувачі ІКМ, внутрішні системи забезпечення якості надання послуг, тощо).

C_3 – *Серверний кластер.* Клас описує фізичні об'єкти, що є складовими частинами розподіленої обчислювальної системи. Атрибутами серверного кластера є кількість його вузлів, завантаженість та модель кластера як СМО: $C_3 = (A_3, D_3)$.

C_4 – *Сервер.* Цей клас описує фізичні об'єкти, що є складовими частинами серверного кластера. Атрибутами сервера э його модель енергоспоживання та обчислювальні ресурси: $C_4 = (A_4, D_4)$.

C_5 – *Обчислювальний ресурс.* До цього класу відносяться поняття, що описують фізичні обчислювальні ресурси сервера. Атрибутом обчислювального ресурсу є його обсяг: $C_5 = (A_5, D_5)$.

C_6 – *Оперативна пам'ять.* Цей клас відображає поняття фізичного ресурсу оперативної пам'яті сервера. Клас має успадкований атрибут "обсяг", що відображає обсяг доступної оперативної пам'яті сервера V_{RAM_j}: $C_6 = (A_6, D_6)$.

C_7 – *Процесорне ядро.* Клас описує поняття фізичного процесорного ресурсу сервера. Він має успадкований атрибут "обсяг", що відображає кількість обчислювальних ядер процесора сервера k_{core_j}: $C_7 = (A_7, D_7)$.

C_8 – *Накопичувач даних.* За допомогою цього класу відображається поняття фізичного ресурсу – накопичувача даних (жорсткого диску, SSD-диску, тощо). Клас має успадкований атрибут "обсяг", що відображає доступний обсяг накопичувача сервера: $C_8 = (A_8, D_8)$.

C_9 – *Навантаження.* Цей клас відповідає за опис поняття вхідного навантаження обчислювальної системи. Він характеризується атрибутом "статистична модель", що відповідає поняттю статистичної кривої вхідного навантаження: $C_9 = (A_9, D_9)$. Наслідують цей клас окремі класи для різних типів навантаження інформаційно-комунікаційної мережі, що розглядаються, а саме: навантаження SDN та NFV, Network Slicing, Edge Computing та bDDN. Ці типи навантаження визначають вимоги SLA щодо їх обслуговування.

C_{10} – *Обчислювальна робота*. Цей клас описує поняття обчислювальної роботи як одиниці вхідного навантаження в системі. Клас має атрибут "об'єм", що

виражає обсяг обчислень, необхідних для обробки роботи (наприклад, у кількості елементарних операцій): $C_{10} = (A_{10}, D_{10})$.

C_{11} – *Вимога.* Цей клас описує вимоги обчислювальних робіт до фізичних ресурсів системи. Вимога має ідентифікатор роботи у якості атрибута для прив'язки конкретної вимоги до роботи, яка її висуває: $C_{11} = (A_{11}, D_{11})$.

C_{12} – *Максимально допустимий час в обробці*. Описує одну із вимог обчислювальної роботи. Це час Δt_{max_i}, по проходженню якого робота виводиться з процесу обробки, навіть якщо обробку не було успішно завершено. Клас наслідує атрибут "ідентифікатор роботи" для прив'язки конкретної вимоги до роботи, яка її висуває: $C_{12} = (A_{12}, D_{12})$.

C_{13} – *Мінімальний об'єм пам'яті*. Клас описує одну із вимог обчислювальної роботи, що відображає мінімально допустимий обсяг оперативної пам'яті сервера $V_{RAM_{\min i}}$, при якому робота іще може надходити на обробку на цьому сервері. Клас наслідує атрибут "ідентифікатор роботи" для прив'язки конкретної вимоги до роботи, яка її висуває: $C_{12} = (A_{13}, D_{13})$.

C_{14} – *Кількість ядер*. Клас описує одну із вимог обчислювальної роботи, що відображає мінімально допустиму кількість вільних процесорних ядер сервера $k_{cores_{\min i}}$, що будуть відведені під обробку даної роботи. Клас наслідує атрибут "ідентифікатор роботи" для прив'язки конкретної вимоги до роботи, яка її висуває: $C_{14} = (A_{14}, D_{14})$.

C_{15} – *Мінімальний об'єм вільного сховища*. Клас описує одну із вимог обчислювальної роботи, що відображає мінімально допустимий об'єм вільного сховища сервера $V_{storage_{\min i}}$, що бути використаний в процесі обробки. Клас наслідує атрибут "ідентифікатор роботи" для прив'язки конкретної вимоги до роботи, яка її висуває: $C_{15} = (A_{15}, D_{15})$.

C_{16} – *Вимоги SLA*. Клас описує поняття SLA як угоди між користувачем та постачальником послуг розподіленої обчислювальної системи, щодо рівня надання послуг. Клас має атрибут "коефіцієнт готовності" $p_{avail_{SLA}}$, що обмежує знизу коефіцієнт готовності системи: $C_{16} = (A_{16}, D_{16})$.

C_{17} – *Енергоефективність.* Описує поняття енергоефективності обчислювальної системи, що виражається за допомогою запропонованого показника енергоефективності E_{Σ} (обсяг навантаження обробленого системою з N вузлів при споживанні обсягу енергії W).

C_{18} – *Продуктивність.* Описує поняття продуктивності обчислювальної системи C_{Σ} в цілому як обсяг обчислювального навантаження обробленого системою за одиницю часу.

C_{19} – *Програмне забезпечення обробки даних.* Відображає поняття програмного забезпечення, що використовується у розподіленій обчислювальній системі для розподілу та обслуговування навантаження, керування ресурсами системи обробки, забезпечення відмовостійкості та надійності системи, тощо.

C_{20} – *Брокер розподілу навантаження.* Клас описує складову частину програмного забезпечення обробки даних, що відповідає за розподіл навантаження між вузлами обчислювальної системи.

C_{21} – *Блок моніторингу та прогнозування навантаження.* Клас описує складову частину програмного забезпечення обробки даних, що відповідає за моніторинг та прогнозування навантаження у системі.

C_{22} – *Блок розподілу та масштабування ресурсів.* Клас описує складову частину програмного забезпечення обробки даних, що відповідає за розподіл та масштабування фізичних ресурсів у системі.

C_{23} – *Блок атестації АЗ.* Клас описує складову частину програмного забезпечення обробки даних, що відповідає за атестацію та контроль стану АЗ системи розподіленої обробки даних.

C_{24} – *База даних.* Клас описує поняття представлення даних в організованій формі для їх ефективного зберігання.

Асоціативні відношення: $\boldsymbol{R_{AS} = \{C_i\ X\ C_j\}}$

"використовує" – відображає зв'язок між класом *Джерело навантаження* та *Розподілена обчислювальна система* та показує пряме або опосередковане використання користувачем обчислювальної системи або її частин;

“*надсилає на обробку*” – зв’язок показує відношення між класом *Джерело навантаження та Обчислювальна робота;*

“*створює*” – відображає опосередкований зв’язок між класами *Джерело навантаження та Навантаження*;

“*забезпечує виконання*” – цей зв’язок описує відношення між *Розподіленою обчислювальною системою* та *Вимогами SLA*. Система повинна працювати таким чином, щоб забезпечити виконання вимог SLA;

“*оцінюється для*” – показує зв’язок між параметрами обчислювальної системи (*Енергоефективність* та *Продуктивність*) та самою *Розподіленою обчислювальною системою;*

“*визначає*” – цей зв’язок описує відношення між класом *Обчислювальний ресурс* та *Продуктивність*. Суть цього зв’язка полягає у відображенні впливу кількості та якості обчислювальних ресурсів сервера на його продуктивність

“*обробляє*”- зв’язок, встановлений між класами *Сервер* та *Обчислювальна робота,* та відповідно описує процес обробки роботи на сервері

“*має*” – зв’язок, що показує логічну приналежність сутності одного класу до іншого. Цей зв’язок мають класи *Сервер-Обчислювальний ресурс, Сервер – Програмне забезпечення обробки даних, Обчислювальна робота – Вимога*

“*визначають вибір*” – описує зв’язок між класами *Вимога та Сервер*. Фізична сутність цього зв’язка – вимоги обчислювальної роботи визначать підмножину серверів, на яких може оброблятись дана робота

Відношення “частина-ціле”

Відношення „частина-ціле” визначаються між класами „*Розподілена система обробки даних*” та „*Серверний кластер*”, „*Серверний кластер*” та „*Сервер*”, „*Навантаження*” та „*Обчислювальна робота*”, щоб показати, що одна сутність є частиною іншої.

Запропонована онтологічна модель дозволяє систематизувати та якісно описати складні взаємозв’язки між показниками ефективності досліджуваного процесу обслуговування навантаження в інформаційно-комунікаційній мережі та параметрами, що впливають на них. Однак, постає необхідність кількісного опису

цих взаємозв’язків та кожного з впливаючих параметрів. Тому необхідно визначити моделі та методи, які дозволяють кількісно охарактеризувати вплив параметрів (зокрема кількості та характеристик обчислювальних вузлів) на показники ефективності досліджуваного процесу.

2.2 Математична модель системи обслуговування навантаження як системи масового обслуговування

Процес розподіленої обробки обчислювального навантаження може бути описаний в термінах системи масового обслуговування (СМО), з метою отримання математичних виразів для опису зв'язків між параметрами системи та визначеними показниками ефективності та кількісної оцінки цих зв'язків.

2.2.1 Аналіз існуючих математичних моделей систем розподіленого обслуговування навантаження

Розглянемо досліджувану розподілену обчислювальну систему із точки зору представлення її у вигляді СМО, спершу проаналізувавши існуючі в літературі моделі схожих систем.

Авторами роботи [103] була запропонована комплексна модель процесу обслуговування навантаження у гетерогенному ЦОД як системи масового обслуговування. Автори пропонують представити систему у вигляді двох СМО, що взаємодіють між собою: перша СМО охоплює опис процесу надходження заявок від користувачів до центрального брокера, що здійснює розподіл навантаження між серверами, а друга – процес обробки заявок на кожному із серверів, які в свою чергу є багатоканальними СМО (роль каналів виконують обчислювальні ядра процесора). Запропоновану систему зображено на рисунку (Рисунок 2.3).

Першу СМО із сервером у вигляді брокера автори представили у вигляді одноканальної СМО зі скінченною чергою, найпростішим вхідним потоком запитів та експоненціальним розподілом часу обробки запитів *(M/M/1/k)*, у той час як друга СМО з каналами у вигляді обчислювальних ядер мала вигляд багатоканальної СМО

з пуассонівським вхідним потоком та без втрат *(M/M/C$_i$)*. Основним недоліком такої системи є те, що автори не розглядають можливість обробки одного запита користувача на багатьох процесорах, тобто не розглядають можливе розпаралелювання задач. Крім того, у запропонованій моделі черги до вузлів обробки (серверів у другій підсистемі СМО) є нескінченними, що призводить до того, що інтенсивність обробки запитів брокером розподілу навантаження ніяк не залежить від завантаженості вузлів обробки (тобто, запит може бути направлено на обробку у будь-який момент часу. У випадку зайнятості сервера запит знаходитиметься у нескінченній черзі перед ним).

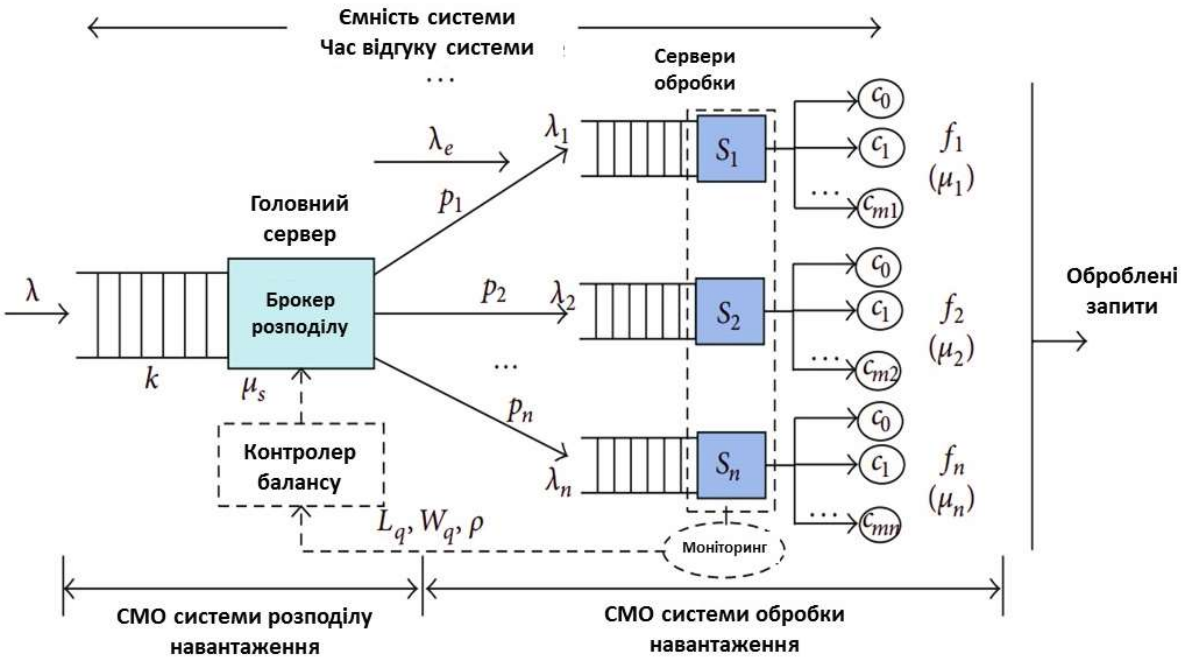


Рисунок 2.3 Дворівнева модель процесу обробки заявок у гетерогенному ЦОД [103]

У роботі [104] запропоновано модель кластера зі скінченної кількості гетерогенних серверів, що мають різну інтенсивність обробки запитів як СМО. У роботі також розглядаються окремі черги до кожного з серверів, при чому довжина цих черг у запропонованій СМО є скінченою. Основним недоліком СМО у дослідженні [104] є те, що автори розглядають лише вхідний пуассонівський потік запитів зі сталою інтенсивністю. Для обчислювальних робіт у рамках навантаження ІКМ інтенсивність надходження зазвичай є змінною та представлена випадковою

величиною. Крім того, автори роботи також не розглядають можливість паралелізації задач, а отже напряму ставлять запит СМО у відповідність обчислювальній роботі, що у випадку застосування паралелізації може не відповідати дійсності.

Автори дослідження [105] запропонували модель серверного кластера як СМО, у якій навантаженням виступають запити користувачів. Вхідне навантаження розподіляється між окремими серверами за допомогою брокера планування навантаження. Автори ввели припущення, що якщо швидкість надходження запитів у кластер дорівнює λ, то швидкість надходження на робочий сервер кластера дорівнює λ_{p_c}, де p_c представляє пропорцію потужності сервера до загальної потужності системи. На кожному із серверів додатково формується окрема черга із запитів. Недоліком даного підходу є те, що автори напряму співставляють запити користувачів із запитами СМО, що значно спрощує процес. Адже у реальній системі один запит користувача може потребувати для своєї обробки більш ніж одне обчислювальне ядро сервера, а сервери в свою чергу можуть мати різну кількість обчислювальних ядер.

Автори роботи [106] замість представлення СМО у вигляді двох частин запропонували багатоканальну модель M/G/k із пуассонівським вхідним потоком, що може бути розділена на k одноканальних систем M/G/1 шляхом декомпозиції [107]. Особливо цікавим у цій роботі є те, що автори врахували можливість зміни інтенсивності обслуговування кожного окремого сервера внаслідок застосування апаратної техніки енергозбереження DVFS, яка застосовується практично в усіх сучасних процесорах [10, 13]. Вони виразили інтенсивність обробки у вигляді формули:

$$\mu' = \mu * \alpha * (\frac{f}{f_{max}}) + \mu * (1 - \alpha)\,, \tag{2.2}$$

де f, f_{max} – поточна та максимальна частота процесора у випадку застосування DVFS; α – коефіцієнт, що показує, наскільки обчислювальна робота є CPU-інтенсивною (CPU – англ. Central Processing Unit – центральний процесор). У дослідженні авторів це значення було прийнято за $\alpha = 0{,}9$, що є типовим для CPU-

інтенсивних навантажень. У джерелах [108, 109] проаналізовано характер потоку запитів до веб-сервера. На основі аналізу статистичних даних автори стверджують, що характер розподілу для вхідного потоку є пуассонівським.

Таким чином, існуючі моделі систем розподіленої обробки навантаження як СМО не враховують того, що інтенсивність надходження запитів для навантаження ІКМ у загальному випадку не є сталою та може різнитись в залежності від часу доби, дня тижня, тощо. Крім того, існуючі дослідження здебільшого не враховують того факту, що при розробці сучасного ПЗ часто застосовуються техніки паралелізації, що призводить до того, що одна обчислювальна робота може вимагати для свого обслуговування $k > 1$ обчислювальних ядер. Виходячи із цього необхідним є створення математичної моделі системи розподіленого обслуговування навантаження ІКМ, що враховує змінний характер інтенсивності навантаження, а також можливість використання технік паралелізації при розробці ПЗ сучасних ІКМ.

2.2.2 Математична модель досліджуваної системи

Опишемо досліджувану у дисертаційній роботі систему у термінах СМО. Результати математичного моделювання досліджуваної ситеми опубліковано у роботі [35].

<u>Моделювання обслуговуючих пристроїв</u>

Природно припустити, що обслуговуючими каналами (або пристроями) виступають сервери (обчислювальні вузли ЦОД у складі ІКМ). Однак:

– на відміну від пристрою СМО, сервер здатний обробляти одночасно декілька обчислювальних робіт за рахунок розміщення їх на різних ядрах процесора (при чому кількість ядер процесора може бути різною для різних серверів);

– обчислювальна робота (код програми) може бути представлена таким чином, що потребуватиме більше одного обчислювального ядра (за рахунок використання механізмів паралельних обчислень).

Ці факти не дозволяють розглядати сервер у якості обслуговуючого пристрою. Натомість, обчислювальне ядро процесора краще підходить для моделювання каналу обслуговування. Можливо вважати, що як і канал СМО, обчислювальне ядро у кожен момент часу знаходиться або у стані “зайняте” або у стані “вільне”. Таким чином, йдеться про багатоканальну СМО з різними каналами, адже інтенсивність обробки запитів μ_j обчислювальними ядрами на різних серверах може відрізнятись.

<u>Моделювання вхідного потоку запитів</u>

Розглянемо характеристику вхідного потоку у досліджуваній системі. Для цього спочатку проаналізуємо характер надходження обчислювальних робіт, а потім – їх співвідношення із запитами, як одиницями навантаження у СМО.

Характер надходження обчислювальних робіт залежить від типу роботи, що розглядається. Так, наприклад, моменти надходження запитів щодо маршрутизації трафіку віртуалізованою мережевою функцією у рамках NFV є випадковими, у той час як обробку Великих даних мережевої статистики, не критичну до часу отримання результатів, можна спланувати так, що наперед відомі обчислювальні роботи надходитимуть на обробку у наперед визначені моменти часу. Те ж саме можна сказати і про інтенсивність надходження робіт: вона залежить від інтенсивності користувацького трафіку і є змінною у випадку навантаження SDN та NFV або Edge Computing, і є контрольованою (тобто сталою на наперед визначений період часу) для не критичних у сенсі часу задач аналізу мережевої статистики або якості надання послуг у рамках концепцій Network Slicing та bDDN. В залежності від типу навантаження, джерелами запитів у системі виступають користувацькі додатки, вузли моніторингу трафіку та параметрів мережі, тощо. Таким чином, у випадку кожної окремої задачі, що вирішується у рамках кожного з розглянутих типів навантаження, доцільним є окремий аналіз параметрів вхідного потоку. Однак, загальна кількість можливих обчислювальних робіт, що потребуються обробки в рамках розглянутих типів навантаження (SDN, NFV, Edge Computing, Network Slicing та bDDN) є дуже великою, що унеможливлює аналіз

характеру вхідного потоку для кожної з них у рамках дисертаційної роботи. Натомість, у рамках математичного моделювання пропонується зупинитись на розгляді найбільш загального випадку вхідного потоку зі змінною інтенсивністю та випадковими моментами надходження взаємно незалежних запитів до системи, обслуговування якого становить найбільшу складність через випадковий характер параметрів потоку. Такі характеристики властиві, наприклад, навантаженню з обчислювальних робіт щодо аналізу відео, що надходять до периферійної хмари (Edge Computing) від пристроїв системи відеоспостереження. Аналіз характеристик та математичне моделювання вхідного потоку для інших типів навантаження пропонується віднести до питань подальших досліджень.

Наступним питанням, що потребує вирішення, є питання моделювання вхідного потоку запитів та поняття запиту в цілому. Чи можна вважати обчислювальну роботу запитом у термінології СМО? Як уже згадувалось, за рахунок використання механізмів розпаралелювання, обчислювальна робота може потребувати для свого виконання більше ніж один обчислювальний пристрій (ядро процесора). Цей факт не дозволяє напряму ставити у відповідність запиту СМО обчислювальну роботу. Натомість, можливим є вважати обчислювальну роботу групою запитів, що надають досліджуваному вхідному потоку запитів властивість неординарності [110]. При цьому, неординарним є як процес надходження запитів, так і процес їх обробки (адже обчислювальна робота, що є групою запитів СМО, надходить до системи, направляється на обробку та виводиться із системи лише як єдине ціле.

Таким чином, розглядається вхідний потік із груп запитів, що надходять до системи у випадкові моменти часу зі змінною інтенсивністю надходження. Кількість груп запитів (обчислювальних робіт) є випадковою величиною із дискретним простором значень та неперервним часом. Оцінимо вхідний потік з точки зору характеристик: відсутності післядії, стаціонарності, ординарності.

Оскільки обчислювальні роботи, що розглядаються, не залежать одна від одної, вважатимемо вхідний потік таким, що має властивість ***відсутності післядії.***

Інтенсивність надходження обчислювальних робіт змінна протягом доби та у різні дні. Так, наприклад, кількість запитів вночі набагато менша, ніж у години найбільшого навантаження, що зазвичай приходяться на вечірні години або середину дня (Рисунок 2.4) [111]. Тому, вхідний потік обчислювальних робіт ***не має властивості стаціонарності.***

Як вже згадувалось, обчислювальна робота являє собою зв'язану групу запитів, що одночасно надходять до системи. Тобто, сам потік обчислювальних робіт є ординарним, оскільки імовірність надходження двох і більше обчислювальних робіт від двох користувачів одночасно дуже низька і її можна вважати рівною нулю. Однак, вхідний потік запитів СМО ***не має властивості ординарності.*** Запити надходять до СМО групами і так само групами обробляються та виводяться із системи.

Таким чином, надалі ми розглядатимемо вхідний потік запитів до досліджуваної системи як ***нестаціонарний неординарний (груповий) пуассонівський потік.***

Характеристикою нестаціонарного потоку є його миттєва густина $\lambda(t)$. Миттєвою густиною потоку називається границя відношення середньої кількості подій, що припадає на елементарний відрізок часу $(t, t + \Delta t)$ до довжини цього відрізка, коли остання прямує до нуля [112]:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{m(t+\Delta t) - m(t)}{\Delta t}, \tag{2.3}$$

де $m(t)$ – математичне очікування кількості подій на відрізку часу $(0; t)$.

Для нестаціонарного пуасонівського потоку (тобто ординарного потоку без післядії, але без властивості стаціонарності) кількість подій на інтервалі часу Δt, що розпочався в момент часу t_0, описується законом Пуассона [112]:

$$p_m(\Delta t, t_0) = \frac{(a^m * e^{-a})}{m!} (m = 0,1,2, \dots), \tag{2.4}$$

де a – математичне очікування кількості подій на інтервалі часу від t_0 до $(t_0 + \Delta t)$, що визначається рівністю:

$$a = \int_{t_0}^{t_0+\Delta t} \lambda(t) dt. \tag{2.5}$$

Аналіз цього виразу показує, що a залежить не лише від довжини інтервалу Δt, але й від його положення на вісі часу t_0. При моделюванні нестаціонарних потоків необхідно знати ***закон зміни інтенсивності надходження*** запитів в часі для періоду, що моделюється. У досліджуваній системі цей закон може бути представлено у вигляді статистичної кривої навантаження на систему протягом доби (Рисунок 2.4).

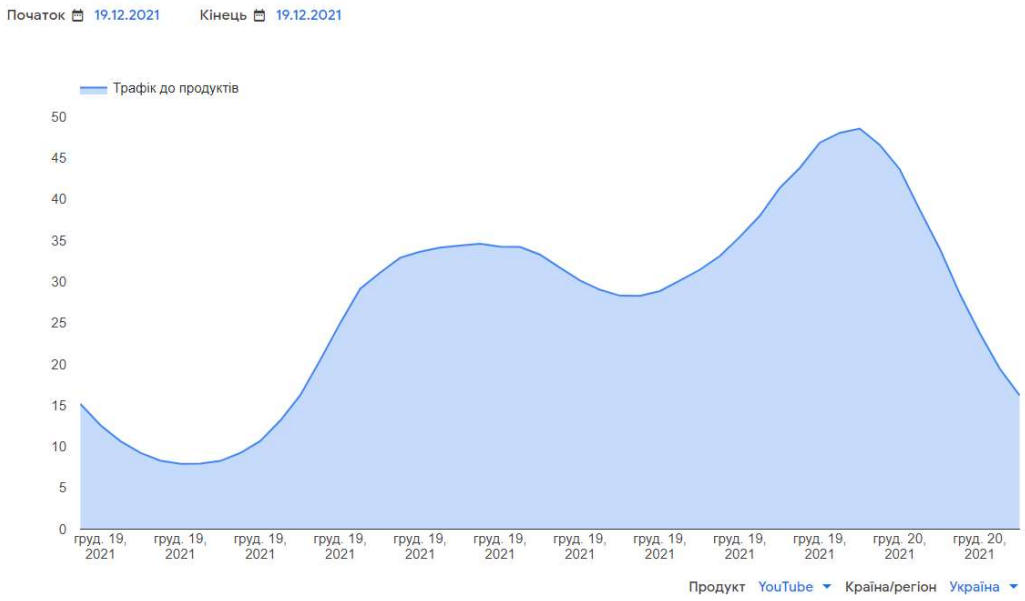


Рисунок 2.4 Крива навантаження як закон зміни інтенсивності надходження запитів нестаціонарного потоку (на прикладі статистики навантаження на відео-сервіс) [111]

Для неординарного вхідного потоку окрім момента надходження запиту необхідно моделювати також кількість запитів, що можуть одночасно надійти до системи. Ця кількість в загальному випадку є також випадковою величиною з певним законом розподілу.

Обчислювальні роботи можуть бути реалізовані у вигляді послідовних інструкцій до одного обчислювального ядра або ***розпаралеленої програми***. Ідея розпаралелювання обчислень полягає в одночасній обробці частин обчислювальної роботи на декількох обчислювальних ядрах [113]. Розрізняють зокрема розпаралелювання задач (коли обчислювальна робота розділяється на декілька незалежних підзадач, що можуть виконуватись окремо на різних ядрах з наступним

об’єднанням результатів) та розпаралелювання за даними (коли однотипні дії виконуються над різними елементами даних одночасно на різних ядрах). Ефективність розпаралелювання задач S визначається законом Амдала (2.6) [113]:

$$S = \frac{1}{v + \frac{1-v}{k_{cores}}}, \tag{2.6}$$

де k_{cores} – кількість процесорів (обчислювальних ядер) у системі, v – кількість операцій, які можуть виконуватись паралельно.

У дисертаційній роботі докладно не розглядається концепції паралельних обчислень, однак вважатимемо обчислювальні роботи такими, що можуть виконуватись як на одному, так і на x ядрах. При цьому вважатимемо необхідну кількість ядер процесора заданою наперед для кожної роботи (з урахуванням розпаралелювання або без нього).

Для моделювання кількості одночасних запитів неординарного вхідного потоку x необхідно визначити закон розподілу, що описує цю випадкову величину. Необхідно зазначити, що кількість одночасних запитів є дискретною випадковою величиною (оскільки обчислювальна робота може виконуватись на $x \in \mathbb{Z}$ ядрах, при чому значення x є обмеженим зверху максимально можливою кількістю ядер процесора одного сервера X_{max}).

Для однозначного визначення розподілу x необхідно проаналізувати реалізацію набору задач у конкретній системі з точки зору використання технік розпаралелювання (якою в середньому є частка не розпаралелених задач, яка частка задач розпаралелена та оптимізована для обробки на наперед визначеній кількості ядер, та яка частка задач розпаралелена та здатна виконуватись на максимально доступній кількості ядер в системі). Прогнозування розподілу для загального випадку обчислювальних робіт, що формують навантаження ІКМ, є складною задачею, а також не є можливим стверджувати, що у найближчому майбутньому усі обчислювальні роботи будуть розпаралелені, оскільки розпаралелювання обчислень значно ускладнює розробку програм, хоча для деяких задач воно є необхідним. Наприклад, одним із яскравих прикладів обчислювальної роботи, що значно виграє від розпаралелювання є аналітика відео у периферійній хмарі Edge

Computing (зокрема у системах відеоспостереження), що основана на машинному навчанні [114]. Згідно з твердженням авторів роботи [114], для збільшення частки розпаралелених програм у сфері мобільних хмарних обчислень необхідна розробка проміжного програмного забезпечення (англ. Middleware), що дозволить абстрагувати тонкощі паралельного програмування. На сьогоднішній день уже існує ряд підходів націлених на спрощення розробки розпаралелених додатків у сфері мобільних хмарних обчислень [115, 116], які можуть відкрити шлях для розробки паралельних додатків зокрема для периферійних хмар у системах 5G. Однак, деякі задачі не є доцільним розпаралелювати з огляду на те, що задача розпаралелювання є досить складною оптимізаційною задачею та потребує додаткових витрат ресурсів на планування та розробку.

Для простоти подальших математичних перетворень введемо допущення, що розподіл кількості ядер, на яких повинна виконуватись обчислювальна робота (що одночасно є розподілом кількості запитів неординарного вхідного потоку), у досліджуваній системі є ***рівномірним***. Однак, у загальному випадку, аналогічні розрахунки можуть бути проведені для будь-якого іншого типу розподілу та відповідного йому математичного очікування кількості запитів неординарного вхідного потоку.

У випадку рівномірного розподілу імовірності надходження до системи обчислювальної роботи, що:

– не є розпаралеленою (виконується на 1 ядрі);

– розпаралелена для виконання на 2 ядрах;

– …

– розпаралелена для виконання на x ядрах ($x \in \mathbb{Z}, x \leq X_{max}$);

є рівними, тобто: $p(x = 1) = p(x = 2) = \ldots = p(x = X_{max})$. А функція розподілу кількості одночасних запитів для досліджуваного неординарного вхідного потоку має вигляд:

$$F(x) = p(x \leq X) = \begin{cases} 0, x < 1 \\ \frac{x}{X_{max}}, 1 \leq x \leq X_{max} \\ 0, x > X_{max} \end{cases} \quad (2.7)$$

та графічно зображена на рисунку (Рисунок 2.5).

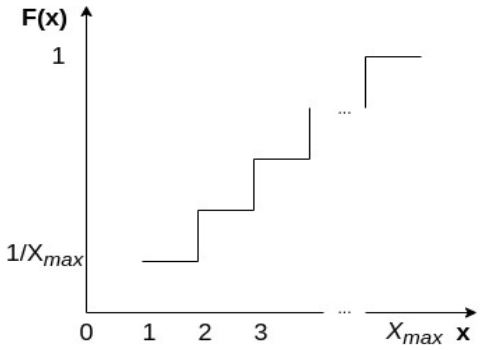


Рисунок 2.5 Функція розподілу кількості одночасних запитів неординарного вхідного потоку

<u>Характеристики процесу обслуговування</u>

Визначимо ***дисципліну обслуговування*** у досліджуваній системі. В залежності від правил, що застосовуються до нових запитів у випадку зайнятості усіх обслуговуючих пристроїв, розрізняють СМО з втратами (запит, що надійшов до СМО в момент зайнятості усіх пристроїв миттєво покидає систему), СМО з безкінечною чергою та змішаний тип СМО (з кінцевою довжиною черги та втратами). У досліджуваній системі обчислювальні роботи спершу надходять до вузла-брокера, який розподіляє роботи за певними правилами між обчислювальними вузлами системи (у найпростішому випадку це правило FIFO (first in first out або перший прийшов, перший обслужився)). Таким чином, на вузлі-брокері формується черга з обчислювальних робіт, що очікують обробки. В загальному випадку ця черга обмежена здатностями апаратного та програмного забезпечення розподілу навантаження. Позначимо цю довжину черги Q. При цьому, Q обчислювальних робіт, що містяться у черзі є групами запитами, кожна з яких складаються з x запитів СМО (у даному випадку, згідно з припущенням, x – це рівномірно розподілена випадкова величина). При переході від обчислювальних

робіт до запитів довжина черги стає рівною $Q * M(x)$, де $M(x)$ – математичне очікування випадкової величини x. Таким чином, йдеться про СМО змішаного типу зі скінченною чергою та втратами.

Обробка запитів також є груповою, тобто запити надходять в обробку групами по x запитів і так само групами покидають систему. Вважатимемо закон розподілу часу обробки запитів експоненціальним [103]. Інтенсивність обробки кожного каналу обслуговування (обчислювального ядра процесора) в загальному випадку різна та рівна μ_j.

Таким чином ***математичною моделлю*** процесу розподіленого обслуговування навантаження у досліджуваній системі можна вважати багатоканальну СМО зі скінченною чергою довжини $Q * M(x)$ (де x – випадкова величина з математичним очікуванням $M(x)$, що відповідає кількості запитів в одній обчислювальній роботі) та з втратами. Вхідний потік запитів в загальному випадку є нестаціонарним неординарним пуассонівським потоком зі змінною інтенсивністю $\lambda(t)$ та випадковою величиною кількості запитів x, що одночасно надходять до системи (згідно з введеним допущенням, величина x має рівномірний розподіл). Джерело запитів нескінченне. Обслуговуючі пристрої представлені ядрами процесорів у системі та мають різну інтенсивність обробки μ_j. Обчислювальна робота формує групу запитів. Кожен запит займає один обслуговуючий пристрій. При цьому, потік обчислювальних робіт є ординарним (на відміну від потоку запитів). Графічне представлення такої системи наведено на рисунку (Рисунок 2.6).

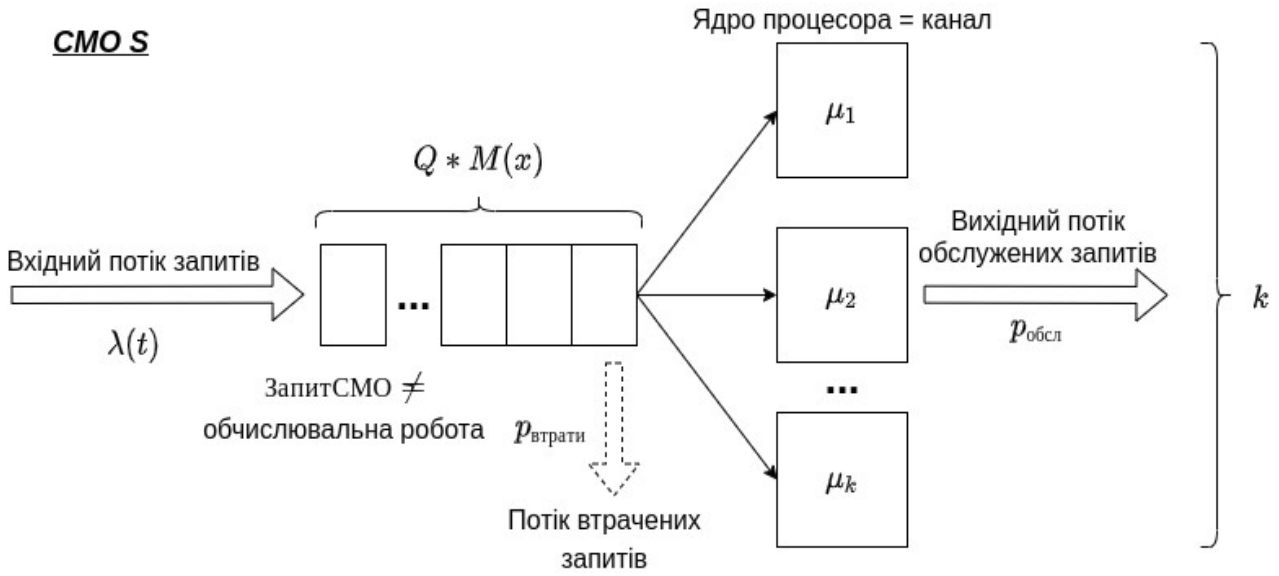


Рисунок 2.6 Графічне представлення досліджуваної системи як СМО

2.2.2.1 Метод перетворення моделі з метою спрощення розрахунків

За рахунок властивостей нестаціонарності та неординарності визначити показники ефективності такої системи (середній час в обробці, імовірність втрати запиту, тощо) за стандартними формулами для найпростішого вхідного потоку не вдається. Тому, з метою переходу до простіших розрахунків, пропонується така методика перетворення математичної моделі:

1. Перехід до стаціонарного неординарного потоку за рахунок представлення кривої інтенсивності надходження обчислювальних робіт до системи у вигляді дискретної функції;

2. Застосування переходу до комплектів серверів для визначення втрат при неординарному потоці заявок.

У процесі спрощення моделі особливу увагу приділено збереженню її точності та адекватності реальному об’єкту дослідження. При цьому свідомо допускаються певні втрати точності, але зазначаються параметри трансформованої моделі, що впливають на її точність.

Як вже згадувалось, інтенсивність вхідного потоку запитів є функцією часу $\lambda(t)$ і в загальному випадку ця функція є неперервною. Однак, із певним наближенням можна представити цю функцію у вигляді дискретної функції $\lambda^D(t)$,

при чому, величина кроку дискретизації впливатиме на точність наближених розрахунків та адекватність моделі реальному об'єктові відповідно (Рисунок 2.7). Для такої функції при прийнятому кроці Δt її значення у будь-який момент часу на відрізку $(t_0, t_0 + \Delta t]$ є константою та дорівнює середньому значенню на даному відрізку:

$$\lambda^t = \lambda^D(\mathrm{t}|\mathrm{t} \in (t_0, t_0 + \Delta t]) = const = \frac{\int_{t_0}^{t_0+\Delta t} \lambda(t)dt}{\Delta t}. \qquad (2.8)$$

Таким чином, надалі ми можемо розглядати стаціонарний неординарний вхідний потік на кожному з інтервалів Δt, інтенсивність якого дорівнює середньому значенню функції $\lambda(t)$ на відповідному відрізку (що дорівнює λ^t), а модель є значно простішою. При чому, розмір інтервалів дискретизації Δt повинен бути вибраним достатньо малим для мінімізації втрати точності. Проблему визначення розміру інтервалів дискретизації докладно розглянуто у підпункті 2.2.2.2

Рисунок 2.7 Приклад представлення $\lambda(t)$ (синя крива) дискретною функцією $\lambda^D(t)$ (зелена ламана) з метою позбавлення від нестаціонарності вхідного потоку (на прикладі статистики навантаження на відео-сервіс) [111]

Особливості роботи зі стаціонарним неординарним потоком було розглянуто зокрема у роботі [117], де було запропоновано перейти від неординарного потоку

до ординарного шляхом переходу до розгляду комплектів каналів (пристроїв обслуговування), кожен з яких обробляє один груповий запит. При сталій кількості запитів у групі такий перехід здійснюється шляхом ділення кількості серверів на кількість запитів у групі. Як зазначено у роботі [117], вхідний потік при цьому залишається пуассонівським.

Однак, як уже згадувалось, у досліджуваній системі кількість запитів у груповому потоці не є сталою. Натомість вона є випадковою величиною. Тому, при переході до комплектів серверів доцільно розглядати математичне очікування випадкової величини як дільник для перетвореної кількості серверів. Таким чином, якщо у досліджуваній системі S кількість каналів (ядер процесора) була рівною k, то у перетвореній системі S' кількість каналів буде рівною $k' = k/M(x)$, де $M(x)$ – математичне очікування кількості одночасних запитів x у групі. Для рівномірно розподіленої випадкової величини математичне очікування рівне: $M(x) = \frac{X_{max}+X_{min}}{2}$.

Таким чином, отримуємо перетворену СМО, в якій вхідний потік запитів є ординарним та стаціонарним на кожному із проміжків часу Δt. По суті, перетворена система S' являє собою систему із запитами, що відповідають ніби-то “однаковим обчислювальним роботам” розпаралеленим для виконання на $M(x)$ ядрах. Обчислювальними каналами в такій перетвореній системі є уявні сервери з однаковою кількістю ядер $M(x)$. Визначимо кількісні характеристики системи S'.

Оскільки визначено інтенсивність надходження запитів на інтервалі Δt як $\lambda^t = \lambda^D(t|\mathrm{t} \in (t_0, t_0 + \Delta t])$, то при переході до перетвореної системи S' отримуємо інтенсивність надходження відповідних комплектів запитів рівну $\lambda' = \frac{\lambda^t}{M(x)} = \frac{\int_{t_0}^{t_0+\Delta} \lambda(t)dt}{\Delta t * M(x)}$. Довжина черги у перетвореній системі є рівною $Q' = \frac{Q*M(x)}{M(x)} = Q$, що статистично співпадає із довжиною черги обчислювальних робіт. Потік вхідних запитів у такій системі є пуассонівським, а розподіл часу обробки запитів – експоненціальним. Каналами обслуговування виступають k'

груп з $M(x) = \frac{X_{max}+X_{min}}{2}$ ядер процесора (у випадку рівномірного розподілу x). Інтенсивність обслуговування для кожного окремого ядра у нашій постановці задачі була рівна μ_j. При переході до системи S', сумарна інтенсивність i-го комплекту каналів відповідно буде дорівнювати $\mu_i{}' = \sum_{j=1}^{M(x)} \mu_j$ запитів в одиницю часу. В загальному випадку інтенсивність обробки μ_j для кожного обчислювального ядра є різною та визначається фізичними параметрами ядра. Однак, при розгляді комплектів ядер, сумарну інтенсивність кожного комплекту із деякою похибкою можна вважати сталою та такою, що дорівнює середньому значенню суми інтенсивностей в рамках кожного комплекту:

$$\mu' = \frac{\sum_{i=1}^{k'} \mu_i{}'}{k'} = \frac{\sum_{i=1}^{k'} \sum_{j=1}^{M(x)} \mu_{ij}}{k'}, \tag{2.9}$$

де μ_{ij} – середня інтенсивність обробки для j-го ядра i-го комплекту каналів. Графічне представлення перетвореної СМО S' наведено на рисунку (Рисунок 2.8).

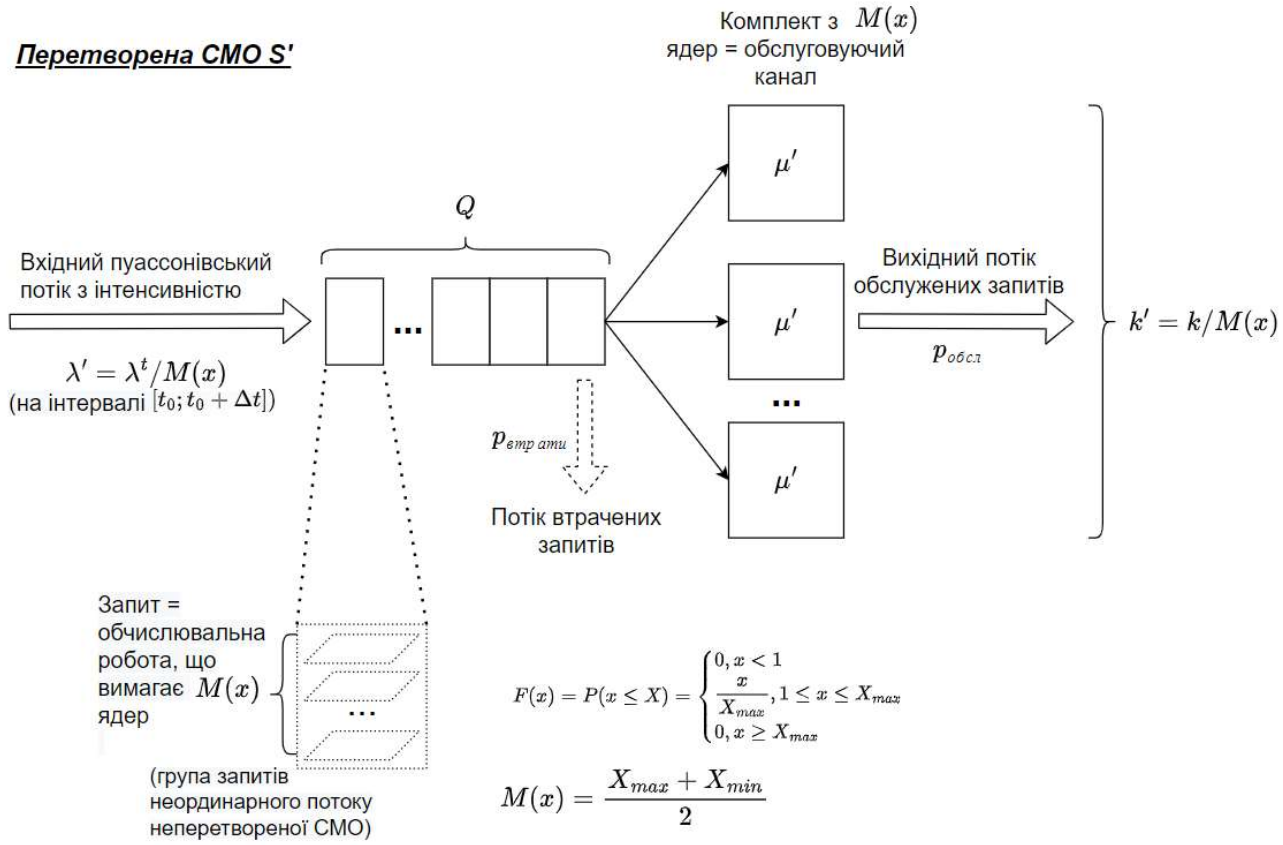


Рисунок 2.8 Графічне представлення перетвореної СМО S'

Аналізуючи отриману систему, можливо розглядати деяку скінченну кількість станів СМО S', при чому перехід з одного стану в інший здійснюється під дією подій надходження та обробки запитів. Потоки надходження та обробки запитів у перетвореній системі S' на інтервалі часу Δt можна вважати найпростішими, що дає нам змогу представити цю систему у вигляді марківського процесу. Граф станів такого процесу зображено на рисунку (Рисунок 2.9), де множина станів: S_0 – в системі немає жодного запиту, всі канали обслуговування вільні, черги немає; S_1 – в системі 1 запит, 1 канал обслуговування зайнятий відповідно, черги немає; S_2 – в системі 2 запити, 2 канали обслуговування зайняті відповідно, черги немає;…; $S_{k'}$ – в системі k' запитів, всі канали обслуговування зайняті, черги немає; $S_{k'+1}$– в системі $(k'+1)$ запит, всі канали зайняті, один запит очікує в черзі;...; $S_{k'+Q}$ – в системі $(k'+Q)$ запитів, всі канали та місця в черзі зайняті.

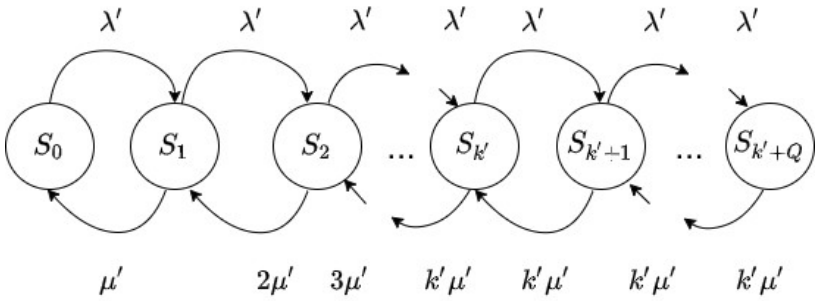


Рисунок 2.9 Граф станів для СМО S'

Запишемо систему диференційних рівнянь Колмогорова для імовірностей станів за умови, що у початковий момент часу система знаходилась у стані S_0:

$$\begin{cases} p_0'(t) = -\lambda' p_0(t) + \mu' p_1(t) \\ p_1'(t) = \lambda' p_0(t) - (\lambda' + \mu') p_1(t) + 2\mu' p_2(t) \\ p_2'(t) = \lambda' p_1(t) - (\lambda' + 2\mu') p_2(t) + 3\mu' p_3(t) \\ \dots \\ p_j'(t) = \lambda' p_{j-1}(t) - (\lambda' + j\mu') p_j(t) + (j+1)\mu' p_{j+1}(t) \\ \dots \\ p_{k'}'(t) = \lambda' p_{k'-1}(t) - (\lambda' + k'\mu') p_{k'}(t) + k'\mu' p_{k'+1}(t) \\ p_{k'+1}'(t) = \lambda' p_{k'}(t) - (\lambda' + k'\mu') p_{k'+1}(t) + k'\mu' p_{k'+2}(t) \\ \dots \\ p_{k'+Q}'(t) = \lambda' p_{k'+Q-1}(t) + k'\mu' p_{k'+Q}(t) \\ \sum_{i=0}^{k'+Q} p_i(t) = 1 \\ p_0(0) = 1, p_1(0) = 0, p_2(0) = 0, \dots, p_{k+Q}'(0) = 0 \end{cases} \tag{2.10}$$

У граничному стаціонарному режимі роботи (при $t \to \infty$) система рівнянь набуває вигляду:

$$\begin{cases} -\lambda' p_0(t) + \mu' p_1(t) = 0 \\ \lambda' p_0(t) - (\lambda' + \mu') p_1(t) + 2\mu' p_2(t) = 0 \\ \lambda' p_1(t) - (\lambda' + 2\mu') p_2(t) + 3\mu' p_3(t) = 0 \\ \dots \\ \lambda' p_{j-1}(t) - (\lambda' + j\mu') p_j(t) + (j+1)\mu' p_{j+1}(t) = 0 \\ \dots \\ \lambda' p_{k'-1}(t) - (\lambda' + k'\mu') p_{k'}(t) + k'\mu' p_{k'+1}(t) = 0 \\ \lambda' p_{k'}(t) - (\lambda' + k'\mu') p_{k'+1}(t) + k'\mu' p_{k'+2}(t) = 0 \\ \dots \\ \lambda' p_{k'+Q-1}(t) + k'\mu' p_{k'+Q}(t) = 0 \\ \sum_{i=0}^{k'+Q} p_i(t) = 1\,. \end{cases} \tag{2.11}$$

Ця система представляє так звані граничні імовірності станів системи. В результаті вирішення системи, позначивши для зручності $\rho = \lambda'/\mu'$ отримуємо значення для граничних імовірностей:

$$p_0 = [1 + \frac{\rho}{1!} + \frac{\rho^2}{2!} + \dots + \frac{\rho^{k'}}{k'!} + \dots + \frac{\rho^{k'+1} * \left(1 - \left(\frac{\rho}{k'}\right)^Q\right)}{k' * k'! * \left(1 - \frac{\rho}{k'}\right)}]^{-1},$$

$$p_j = \frac{\rho^j}{j!} p_0, (j = 1, \dots, k'), \tag{2.12}$$

$$p_{k'+r} = \frac{\rho^{k'+r}}{k'^r * k'!} * p_0, (r = 1, \dots, Q).$$

Таким чином, отримано ***математичну модель*** системи розподіленого обслуговування навантаження ІКМ у вигляді СМО зі скінченною чергою та з втратами, що описується параметрами:

– $k' = k/M(x)$ – кількість обслуговуючих каналів, де $k = k_{cores}$ – задана за умовою кількість обчислювальних ядер у системі; $M(x)$ – математичне очікування кількості одночасних запитів x у групі. Для рівномірно розподіленої випадкової величини математичне очікування рівне $M(x) = \frac{X_{max}+X_{min}}{2}$; X_{max} – максимальна кількість ядер процесора одного сервера у досліджуваній системі (що відповідає максимально можливій кількості ядер, яку може вимагати обчислювальна робота для обробки); $X_{min} = 1$ – мінімально можлива кількість ядер, яку може вимагати обчислювальна робота для обробки.

– $\lambda' = \frac{\int_{t_0}^{t_0+\Delta} \lambda(t)dt}{\Delta t * M(x)}$ – інтенсивність надходження запитів на відрізку часу $[t_0; t_0 + \Delta t]$, де $\lambda(t)$ – задана за умовою крива вхідного навантаження.

– $\mu' = \frac{\sum_{i=1}^{k\prime} \sum_{j=1}^{M(x)} \mu_{ij}}{k\prime}$ – середня інтенсивність обробки заявок, де $\mu_{ij} = \bar{\mu}_j$ – середня інтенсивність обробки j-го обчислювального ядра.

– $Q' = \frac{Q * M(x)}{M(x)} = Q$ – довжина черги (співпадає із заданою довжиною черги).

Основними показниками ефективності СМО зі скінченною чергою та з втратами, формальні вирази для яких можна отримати із формул Літтла [118], є:

– Імовірність втрати запиту:

$$p_{loss} = p_{k\prime+Q} = \frac{\rho^{k\prime+Q}}{k\prime^{Q} * k\prime\,!} p_0. \tag{2.13}$$

– Абсолютна пропускна здатність:

$$A = \lambda' \left(1 - \frac{\rho^{k\prime+Q}}{k\prime^{Q} * k\prime\,!} p_0\right). \tag{2.14}$$

– Середня кількість заявок у черзі:

$$L_Q = \frac{\rho^{k\prime+1} p_0 (1 - (Q + 1 - Q\frac{\rho}{k'})(\frac{\rho}{k'})^Q}{k' * k'!\,(1 - \frac{\rho}{k'})^2}. \tag{2.15}$$

– Середня кількість заявок в обробці, що відповідає середній кількості зайнятих каналів обслуговування:

$$\bar{n} = \rho\left(1 - \frac{\rho^{k'+Q}}{k'^{Q} * k'!}p_0\right), \tag{2.16}$$

де $\rho = \lambda'/\mu'$.

Нагадаємо, що за своїм фізичним змістом перетворена СМО S' являє собою систему, в якій однакові обчислювальні роботи, кожна з яких вимагає $M(x) = \frac{X_{max}+X_{min}}{2}$ обчислювальних ядер (у випадку рівномірного розподілу x), надходять на обробку на k' однакових обчислювальних серверів із середньою інтенсивністю обробки $\mu' = \frac{\sum_{i=1}^{k\prime}\sum_{j=1}^{M(x)}\mu_{ij}}{k\prime}$ робіт в одиницю часу. Кожен сервер у системі S' при цьому має $M(x)$ обчислювальних ядер. Звісно, така система відрізняється від початкової системи S, в якій сервери та обчислювальні роботи є різними. Однак, при достатньо великій кількості обчислювальних вузлів та обчислювальних робіт ці системи можна вважати аналогічними, що було показано в ході покрокового перетворення системи S до вигляду S'.

На відміну від існуючих, запропонована математична модель враховує змінний характер інтенсивності вхідного навантаження та можливість використання технік паралелізації при розробці ПЗ сучасних ІКМ.

2.2.2.2 Метод дискретизації функції інтенсивності надходження вхідного навантаження

Як вже згадувалось, для спрощення математичної моделі пропонується перехід до стаціонарного неординарного потоку за рахунок дискретизації неперервної кривої інтенсивності надходження обчислювальних робіт. Для цього необхідно визначити середню інтенсивність надходження запитів на кожному з інтервалів Δt за формулою (2.8), при чому величина інтервалу Δt має бути вибрана такою, щоб забезпечити збереження адекватності моделі реальному об’єкту дослідження.

Розглянемо простий штучний приклад відрізка кривої інтенсивності вхідного навантаження (Рисунок 2.10). Припустимо, що для дискретизації функції було обрано сталий інтервал часу Δt, такий що $t_1 - t_0 = t_2 - t_1 = \Delta t$. Для кожного інтервалу було визначено середнє значення інтенсивності $\lambda^{t_1 - t_0}$ та $\lambda^{t_2 - t_1}$(що показані зеленою та червоною прямими на рисунку відповідно).

Як видно з рисунка, абсолютне відхилення екстремумів функції $\lambda(t)$ від середніх значень на інтервалі суттєво відрізняються для інтервалів $t_1 - t_0$ та $t_2 - t_1$, що виражається нерівностями:

$$abs(\lambda_{max}|_{t_2 - t_1} - \lambda^{t_2 - t_1}) >> abs(\lambda_{max}|_{t_1 - t_0} - \lambda^{t_1 - t_0}) \quad (2.17)$$

$$abs(\lambda_{min}|_{t_2 - t_1} - \lambda^{t_2 - t_1}) >> abs(\lambda_{min}|_{t_1 - t_0} - \lambda^{t_1 - t_0}) \quad (2.18)$$

Очевидно, що чим більшою є абсолютна різниця між екстремумами функції та визначеним середнім значенням інтенсивності на інтервалі, тим менш точною є модель вхідного потоку навантаження як стаціонарного. Одним з можливих шляхів підвищення точності моделі є зменшення інтервалу дискретизації Δt. Однак, питання про вибір оптимальної абсолютної величини цього інтервалу при цьому залишається відкритим. Надто малий інтервал дискретизації призводить до зростання накладних витрат на розрахунки (адже на кожному інтервалі ми матимемо окрему модель СМО з різними показниками ефективності, які залежать від інтенсивності надходження запитів на заданому інтервалі).

Таким чином, інтервал дискретизації функції має бути достатньо малим, щоб зазпечити адекватність моделі реальному об’єктові, і при цьому якомога більшим, щоб зменшити накладні витрати на розрахунок необхідних показників ефективності СМО.

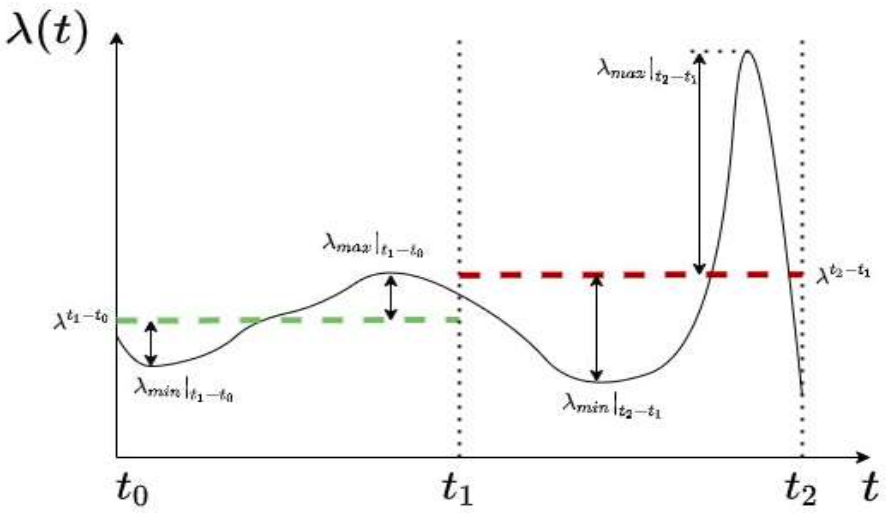


Рисунок 2.10 Дискретизація зі сталим інтервалом часу Δt на приклад двох довільних відрізків кривої інтенсивності вхідного навантаження

Альтернативним підходом є застосування ідеї відомого з радіотехніки підходу квантування за рівнями [119]. Згідно цього підходу, замість вибору сталих проміжків по осі абсцис, вибираються сталі проміжки значень по осі ординат (Рисунок 2.11). Тоді, в межах кроку квантування інтенсивність надходження запитів можна вважати стаціонарною.

Перевагою такого підходу є те, що часові проміжки не є сталими, їх довжина по суті є обернено пропорційною до швидкості зміни функції. Недоліком такого підходу, як і підходу дискретизації, є необхідність правильного визначення кроку квантування. Адже при збільшенні кроку квантування зменшується точність наближення, а при зменшенні, відповідно, зростають накладні витрати на розрахунок.

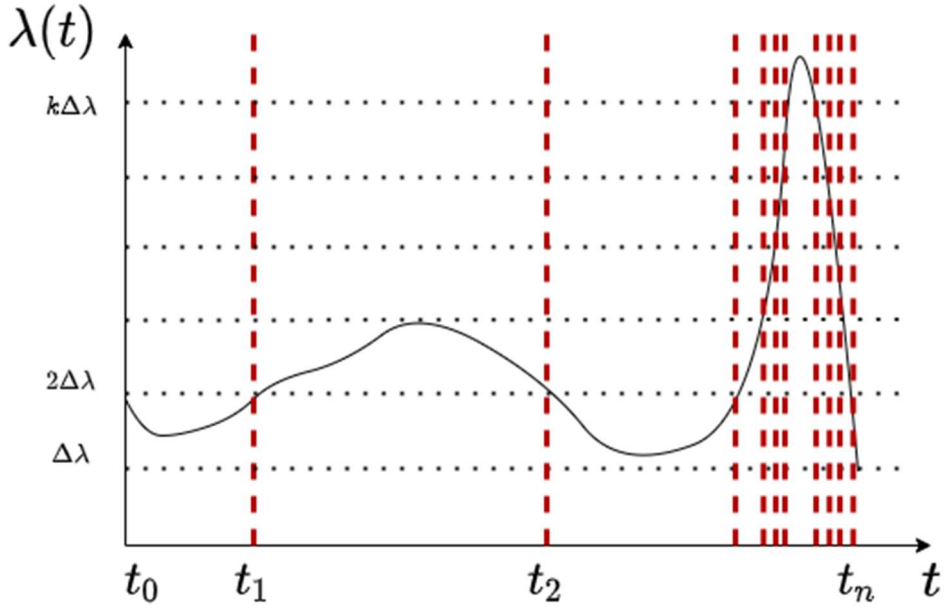


Рисунок 2.11 Застосування ідеї квантування за рівнями для визначення проміжків стаціонарності вхідного потоку запитів $\lambda(t)$

В цілому, крок квантування може визначатися для кожної системи індивідуально на основі експертного аналізу статистичної кривої інтенсивності навантаження. Однак, для повністю автоматизованої системи необхідним є наявність підходу, який дозволить автоматично визначати оптимальний крок квантування для заданої статистичної кривої інтенсивності навантаження.

При виборі підходу перш за все необхідним є врахування фізичного змісту величини інтенсивності навантаження та її впливу на систему. Інтенсивність навантаження у досліджуваній системі S' являє собою кількість запитів, що надходять до системи за одиницю часу. При інтенсивності навантаження $\lambda(t) = 0$ всі обслуговуючі пристрої в системі знаходяться у стані простою, черга вільна. При зростанні $\lambda(t)$ до деякого порогового значення $\lambda(t)_{\text{п}}$ у системі в середньому достатньо одного обслуговуючого пристрою для обслуговування вхідного потоку з урахуванням черги. При подальшому монотонному зростанні $\lambda(t)$ системі потрібно уже 2 обслуговуючі пристрої, і так далі. Тобто, при монотонному зростанні інтенсивності надходження запитів можна зафіксувати певні перехідні моменти, коли середня кількість зайнятих обслуговуючих пристроїв у системі дискретно збільшується. Відповідні порогові величини інтенсивності надходження

$\lambda(t)_{\text{п}}$ можна розрахувати із формули (2.13) для імовірності втрати запиту при заданій допустимій імовірності втрати запиту шляхом розв'язання рівняння (2.13) з одним невідомим λ' при натуральних $k' \in \mathbb{N}^*$ $(k' = 1,2,3,\dots)$. Для цього виразимо λ' із формули (2.13):

$$p_{loss} = \frac{\rho^{k'+Q}}{k'^{Q} * k'!} p_0 = \frac{\left(\frac{\lambda'}{\mu'}\right)^{k'+Q}}{k'^{Q} * k'!} p_0 \Rightarrow$$

$$\Rightarrow \left(\frac{\lambda'}{\mu'}\right)^{k'+Q} = \frac{p_{loss} * k'^{Q} * k'!}{p_0} \Rightarrow \tag{2.19}$$

$$\Rightarrow \lambda' = \mu' * \sqrt[k'+Q]{\frac{p_{loss} * k'^{Q} * k'!}{p_0}}$$

При натуральних $k' \in \mathbb{N}^*$ $(k' = 1,2,3,\dots)$ вираз (2.19) набуває певних дискретних значень, що відповідають пороговим величинам інтенсивності вхідного навантаження $\lambda(t)_{\text{п}}$. Тобто, порогові значення інтенсивності вхідного навантаження, при яких середня кількість необхідних для обслуговування каналів при заданій імовірності втрати запиту p_{loss} дискретно збільшується, пропонується визначати за виразом:

$$\lambda(t)_{\text{п}} = \mu' * \sqrt[k'+Q]{\frac{p_{loss} * k'^{Q} * k'!}{p_0}},\ k' \in \mathbb{N}^*, \tag{2.20}$$

де μ' – середня інтенсивність обробки запитів; k' – кількість обслуговуючих каналів; Q – довжина черги; p_{loss} – задана імовірність втрати запиту.

Тобто, вираз (2.20) дозволяє розрахувати такі значення вхідної інтенсивності, при яких кількості серверів k' стає недостатньо для збереження допустимої імовірності втрати запиту p_{loss}. При послідовному розв'язанні рівняння (2.20) для цілих k' отримаємо набір порогів, що відповідають рівням квантування та відображають "суттєве" для системи зростання інтенсивності вхідного навантаження, яке призводить до необхідності зайняття додаткового обслуговуючого каналу.

2.3 Математична модель енергоспоживання обчислювальних вузлів

Процес енергоефективного обслуговування навантаження в інформаційно-комунікаційній мережі передбачає врахування характеристик енергоспоживання окремих обчислювальних вузлів та системи в цілому у процесі організації обробки. Для оцінки енергоефективності кожного вузла використовується його модель енергоспоживання. Під моделлю енергоспоживання вузла серверного кластера розглядається залежність середньої миттєвої споживаної вузлом потужності від ступеня завантаженості вузла. Залежності енергоспоживання обчислювальних вузлів розподіленої обчислювальної системи від завантаженості їх компонентів може бути визначено програмним шляхом (за допомогою автоматизованих розрахунків) або емпіричним шляхом (за допомогою індивідуальної атестації кожного вузла).

У загальному випадку, сукупне споживання енергії визначається сумарним споживанням усіх компонентів обчислювального вузла (ЦП, елементів оперативної та постійної пам’яті, мережевої карти, тощо). Однак, згідно джерела [120] середній розподіл потужності споживання між структурними елементами сервера становить (у відсотковому співвідношенні): 33% – ЦП, 30% – елементи оперативної пам’яті, 10% – елементи постійної пам’яті, 5% – мережеві елементи, 22% – інше. Тобто, у відповідності до досліджень, проведених в роботі [120], основними споживачами потужності є ЦП та елементи оперативної пам’яті. При цьому, ступінь завантаженості елементів оперативної пам’яті тісно пов’язаний із інтенсивністю використання ЦП. Тому, при побудові моделей енергоспоживання обчислювальних вузлів доцільним є у першу чергу визначати залежність потужності споживання від завантаженості ЦП та відповідної йому завантаженості елементів оперативної пам’яті. Тобто, шукана модель енергоспоживання j-го обчислювального вузла матиме вигляд функції $P_j = f(CPU_j)$, де CPU_j – ступінь завантаженості ЦП j-го обчислювального вузла.

2.3.1 Метод визначення моделей енергоспоживання емпіричним шляхом

Емпіричне визначення моделей енергоспоживання обчислювальних вузлів розподіленої системи полягає у проведенні так званого процесу атестації вузлів системи із подальшою обробкою результатів з метою отримання аналітичних функцій. В рамках процесу атестації значення потужності споживання вузлів визначається безпосередньо у колі живлення кожного сервера. Процес атестації детально описано у пункті 3.1.1 та у публікаціях [42, 43].

Метою даного розділу є опис математичних методів, використаних для обробки результатів атестації. Вибірка, отримана шляхом безпосереднього вимірювання підлягає статистичній обробці. Вважатимемо, що кінцевий результат атестації представлено у вигляді:

$$x = (\tilde{x} \pm \Delta x)_{\alpha};\ \delta\% = \frac{\Delta x}{\tilde{x}} * 100\%, \qquad (2.21)$$

де $\tilde{x}$ – середнє арифметичне значення миттєвих значень вимірюваної величини (напруги або струму відповідно) на інтервалі часу, Δx – абсолютна похибка за формулою розподілу Стьюдента, $\delta\%$ – відносна похибка у відсотках.

Після визначення середніх на інтервалі миттєвих значень струму та напруги у колі живлення j-го обчислювального вузла (при сталому навантаженні ЦП), визначається потужність споживання сервера при досліджуваній завантаженості ЦП j-го обчислювального вузла за базовою формулою: $P = \bar{U} * \bar{I}$, де $\bar{U}$ та $\bar{I}$ – визначені значення напруги та струму відповідно.

Аналогічні дії проводяться для всіх N обчислювальних вузлів досліджуваної системи при всіх M ступенях навантаження ЦП. Результатом обчислень є матриця розмірністю M на N, номери стовпців якої відповідають номерам досліджуваних серверів, а рядки – ступеню навантаженості ЦП, при яких проводяться вимірювання:

$$\begin{array}{l} \\ \text{Завантаженість } CPU\ 0\% \\ \vdots \\ \text{Завантаженість } CPU\ 100\% \end{array} \begin{array}{c} \begin{array}{ccc} \text{Сервер 1} & \dots & \text{Сервер } N \end{array} \\ \begin{pmatrix} P_{11} & \cdots & P_{1N} \\ \vdots & \ddots & \vdots \\ P_{M1} & \cdots & P_{MN} \end{pmatrix} \end{array} \tag{2.22}$$

Для визначення функціональних залежностей потужності споживання серверів від завантаженості їх ЦП на основі емпірично отриманих даних може бути використано ряд відомих математичних методів, зокрема:

– апроксимація лінійною функцією із застосуванням методу найменших квадратів;

– метод інтерполяції поліномом.

Розглянемо особливості кожного із цих методів у контексті розв’язуваної задачі.

2.3.1.1 Апроксимація із застосуванням методу найменших квадратів

Нехай у точках x_i незалежної змінної X отримані виміри $Y = \{y_1, y_2, \dots, y_i, \dots y_n\}$. У рамках дослідження, що проводиться у роботі, незалежними змінними є ступені завантаженості ЦП $x_i = CPU_i$, залежною змінною є значення потужності споживання досліджуваного вузла кластера при рівнях завантаженості x_i.

Задача апроксимації заключається у знаходженні такої функції $f(.)$, що найбільш точно описує залежність змінної Y від змінної X за заданими значеннями в окремих точках (x_i, y_i). При цьому, в загальному випадку функція $f(.)$ може мати як лінійний, так і нелінійний характер (описуватись, наприклад, квадратичною функцією) [121].

При вирішенні задачі побудови моделей енергоспоживання обчсилювальних вузлів часто припускають, що $f(.)$ є лінійною функцією параметрів a, b, тобто має вигляд $f(a, b) = aX + b$ Так, зокрема, у роботі [19] залежність споживаної потужності сервера від завантаженості його ЦП $P = f(CPU)$ вважається лінійною, тобто такою, що описується формулою: $P = a * CPU + b$.

У рамках методу найменших квадратів задача ***апроксимації лінійною функцією*** вирішується шляхом знаходження коефіцієнтів лінійної залежності a та b при яких функція:

$$F(a,b) = \sum_{i=1}^{n} \left(y_i - (ax_i + b)\right)^2 \tag{2.23}$$

приймає найменше значення.

Таким чином мінімізується сума квадратів відхилень експериментально отриманих значень $Y = \{y_1, y_2, \dots, y_i, \dots y_n\}$ від знайденої прямої $f(a,b)$.

Для знаходження коефіцієнтів a та b складається та розв'язується система рівнянь з двома невідомими. Для цього розраховуються та прирівнюються до нуля часткові похідні функції (2.23) за змінними a та b. Отримана система має вигляд:

$$\begin{cases} \dfrac{\partial F(a,b)}{\partial a} = 0 \\ \dfrac{\partial F(a,b)}{\partial b} = 0 \end{cases} \Leftrightarrow \begin{cases} -2\sum_{i=1}^{n}(y_i - (ax_i + b)x_i = 0 \\ -2\sum_{i=1}^{n}(y_i - (ax_i + b) = 0 \end{cases} \Leftrightarrow$$

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} b = \sum_{i=1}^{n} y_i \end{cases} \Leftrightarrow \begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + nb = \sum_{i=1}^{n} y_i \end{cases}. \tag{2.24}$$

Система (2.24) розв'язується будь-яким відомим методом, у результаті чого отримуємо формули для знаходження коефіцієнтів за методом найменших квадратів:

$$\begin{cases} a = \dfrac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \\ b = \dfrac{\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i}{n} \end{cases}. \tag{2.25}$$

При розрахованих таким чином коефіцієнтах a та b сума квадратів відхилень експериментально отриманих значень $Y = \{y_1, y_2, \dots, y_i, \dots y_n\}$ від знайденої прямої $f(a,b)$ має найменше значення. Функція $f(a,b)$ є шуканою лінійною функцією апроксимації.

Апроксимація лінійною функцією дозволяє отримати лише наближений результат, оскільки описувана реальна функція енергоспоживання обчислювальних вузлів не обов'язково має лінійний характер (про це свідчить, наприклад Рисунок 1.4, що зображує дані досліджень [3]). Метод найменших квадратів не обмежується використанням лінійної моделі (альтернативою є використання, наприклад, квадратичної моделі вигляду $f(a,b) = ax^2 + b$). Однак, суттєвим недоліком апроксимації як методу в цілому для вирішення поставленої задачі є той факт, що апроксимація не гарантує проходження отриманої функції через точки y_i. У рамках вирішуваної задачі відхилення функції в опорних точках y_i, що були визначені у рамках експерименту, є небажаним, оскільки вони відіграють суттєву роль показуючи порогові значення потужності споживання при зайнятті $1,2,..,k_{cores_j} \in \mathbb{N}$ ядер процесора.

Для більш точного визначення залежностей енергоспоживання вузлів кластера від їх завантаженості використаємо інструмент інтерполяції, що полягає у пошуку значень всередині обмеженого інтервалу з відомими межами.

2.3.1.2 Метод інтерполяції

Нехай на відрізку $[a,b]$ задана функція $f(x)$. Задача інтерполяції полягає в побудові функції $g(x)$, такої що збігається із заданою $f(x)$ в деякому наборі точок $\{x_1, x_2, \ldots, x_{n+1}\}$ з відрізка $[a,b]$ [121]. Ці точки називаються вузлами інтерполяції. При цьому повинні виконуватись вимоги:

$$g(x_k) = y_k,\ k = 1,2,\ldots,n + 1, \tag{2.26}$$

де y_k – відомі значення функції f(x) в точках x_k. Функція g(x) називається інтерполянтом функції f(x).

Вузли інтерполяції не обов'язково повинні розташовуватися рівномірно на відрізку $[a,b]$.

Якщо $f(x)$ таблична функція, отримана, скажімо, з експерименту, тобто відомі тільки її значення y_k в точках x_k, то, про якість отриманого наближення судити важко. Якість залежить від кількості наявних замірів (точок) на заданому відрізку.

На якість наближення також впливає розташування вузлів інтерполяції, а також гладкість функції $f(x)$.

Розглянемо лінійну інтерполяцію, тобто таку, при якій функція $g(x)$ шукається у вигляді лінійної комбінації деяких функцій:

$$g(x) = \sum_{k=1}^{n+1} a_k \varphi_k(x), \tag{2.27}$$

де для $k = 1,2,\ldots,n + 1$: $\varphi_k(x)$ – задані функції, а a_k – шукані коефіцієнти.

З постановки задачі інтерполяції випливає, що коефіцієнти a_k визначаються шляхом розв'язання системи лінійних алгебраїчних рівнянь:

$$\sum_{k=1}^{n+1} a_k \varphi_k(x_j) = y_j, j = 1,2, \ldots n + 1, \tag{2.28}$$

або в розгорнутій формі:

$$\begin{cases} a_1\varphi_1(x_1) + a_2\varphi_2(x_1) + \cdots + a_{n+1}\varphi_{n+1}(x_1) = y_1 \\ a_1\varphi_1(x_2) + a_2\varphi_2(x_2) + \cdots + a_{n+1}\varphi_{n+1}(x_2) = y_2 \\ \ldots \\ a_1\varphi_1(x_{n+1}) + a_2\varphi_2(x_{n+1}) + \cdots + a_{n+1}\varphi_{n+1}(x_{n+1}) = y_{n+1}. \end{cases} \tag{2.29}$$

Число коефіцієнтів a_k, очевидно, має збігатися з числом вузлів інтерполяції x_k. Це потрібно для того, щоб матриця системи була квадратною (тобто число невідомих збігалося б з числом умов, з яких знаходяться ці невідомі). Крім того, для однозначної розв'язності даної системи (при довільній правій частині) необхідно і достатньо, щоб їх визначник був відмінним від нуля:

$$\begin{vmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \ldots & \varphi_{n+1}(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \ldots & \varphi_{n+1}(x_2) \\ & & \ldots & \\ \varphi_1(x_{n+1}) & \varphi_2(x_{n+1}) & \ldots & \varphi_{n+1}(x_{n+1}) \end{vmatrix} \neq 0 \tag{2.30}$$

Дуже часто в якості системи функцій $\varphi_k(x)$ застосовують поліноми, наприклад ступеня n: $\varphi_1(x) = 1, \varphi_2(x) = x, \varphi_3(x) = x^2, \ldots, \varphi_{n+1}(x) = x^n$.

Тоді відповідний інтерполянт називають інтерполяційним поліномом. Існування і єдиність інтерполяційного полінома гарантується, якщо всі вузли інтерполяції x_k різні. Дійсно, визначник системи лінійних алгебраїчних рівнянь для знаходження коефіцієнтів a_k вигляду:

$$\begin{vmatrix} {x^0}_1 & {x^1}_1 & \dots & {x^n}_1 \\ {x^0}_2 & {x^1}_2 & \dots & {x^n}_2 \\ & & \dots & \\ {x^0}_{n+1} & {x^1}_{n+1} & \dots & {x^n}_{n+1} \end{vmatrix} \tag{2.31}$$

є визначником Вандермонда, який дорівнює $\prod_{i=1,i>j}^{n+1}(x_i - x_j)$ і, отже, є відмінним від нуля за умови, коли всі вузли x_k різні. Оскільки матриця системи невироджена, то розв'язок системи існує і він єдиний. Отже, інтерполяційний поліном існує і він єдиний [122].

Основною перевагою інтерполяції є гарантована точність значення функції в опорних точках, що були визначені емпіричним шляхом. Це є особливо важливим у контексті досліджуваної системи, адже обчислювальні роботи як правило повністю завантажують та звільняють певний дискретний обсяг ресурсів, що відповідає обчислювальним ядрам вузлів. Звичайно, завантаженість одного ядра у процесі обробки обчислювальною роботи також може змінюватись за деяким неперервним законом, однак дискретні значення завантаженості, що відповідають зайнятості $1, 2, 3, .., k_{cores_j} \in \mathbb{N}$ ядер відіграють особливу роль у процесі обробки та повинні бути передані аналітичною функцією із високою точністю.

Тому, надалі у роботі функції $P = f(CPU)$ після визначення у табличному вигляді представляються у вигляді інтерполяційних поліномів ступеня $n = m - 1$ вигляду:

$$P(CPU) = a_0 + a_1 * CPU^1 + a_2 * CPU^2 + \ldots + a_{m-1} * CPU^{m-1}, \tag{2.32}$$

де CPU – ступінь завантаженості ЦП обчислювального вузла.

Показник ступеня визначається на основі даних про проведений дослід (попередню атестацію) в залежності від кількості проведених вимірів. Максимальний ступінь інтерполяційного полінома $n = m - 1$, де m – кількість

вузлів інтерполяції (тобто вимірювань у даному випадку). Теоретично максимальну точність забезпечує многочлен більш високого ступеня [123], тому у роботі прийматимемо n на 1 меншим за кількість виміряних значень потужності споживання при сталому навантаженні на вузол.

Основними перевагами емпіричного визначення моделей енергоспоживання вузлів обчислювальної системи [40, 41] є висока точність та можливість врахування індивідуальних особливостей АЗ. Адже два фізичні обчислювальні вузли, оснащені ідентичними компонентами, можуть не мати абсолютно однакових характеристик потужності споживання (наприклад, через вік, індивідуальні особливості виготовлення компонентів або інші причини). Однак при періодичному застосуванні такий метод є досить затратним за часом та ресурсами, тому що обладнання, яке атестується не може бути використане для одночасної обробки обчислювального навантаження в системі (проведення атестації передбачає тимчасове виведення обчсилювального вузла з нормального процесу обслуговування навантаження в системі).

2.3.2 Метод визначення моделей енергоспоживання програмним шляхом

В основі методу програмного визначення моделей енергоспоживання обчислювальних вузлів лежить розрахунок сумарної потужності споживання вузла як суми потужностей споживання окремих його компонентів. При цьому, використовуються номінальні значення потужності споживання компонентів, що значно спрощує та пришвидшує розрахунок (оскільки необхідність експериментального визначення споживаної потужності відсутня). Основними компонентами обчислювальної системи, потужність споживання яких береться у розрахунок є:

– центральний процесор;

– графічний процесор;

– елементи оперативної пам’яті та материнська плата;

– елементи постійної пам’яті;

– дисплей (за наявності).

Тобто, сумарна потужність споживання обчислювального вузла при кожному розглядуваному рівні завантаженості процесора CPU_j від 0 до 100% розраховується за формулами:

$$P_{\Sigma_j}(CPU_j = 0) = P_{CPU_j|0} + P_{GP\ \ j} + P_{RAM\&MB_j} + P_{HDD_j} + P_{DI\ \ j},$$

$$\ldots \tag{2.33}$$

$$P_{\Sigma_j}(CPU_j = 100) = P_{CPU_j|100} + P_{GPU_j} + P_{RAM\&MB_j} + P_{HDD_j} + P_{DISP_j},$$

де $P_{CP\ \ j|x}$ – потужність споживання ЦП j-го вузла при завантаженості x; $P_{GP\ \ j}$ – потужність споживання графічного процесора j-го вузла; $P_{RAM\&MB_j}$ – потужність споживання елементів оперативної пам’яті та материнської плати j-го вузла; $P_{HD\ \ J}$ – потужність споживання елементів постійної пам’яті; P_{DISP_J} – потужність споживання дисплея (за наявності).

Звичайно, неможливо стверджувати, що ступінь завантаженості усіх компонентів у кожен момент часу є однаковим (так, наприклад, завантаженість ЦП у 100% не означає завантаженість у 100% також і елементів постійної пам’яті). Однак, як вже згадувалось, згідно з джерелом [120], основними споживачами електроенергії у складі обчислювального вузла є ЦП та елементи оперативної пам’яті, ступінь завантаженості яких корелюється. У випадку використання графічного процесора для розрахунків, його споживана потужність також вносить суттєвий внесок у сумарне енергоспоживання вузла, однак виконання обчислювальних робіт на графічному процесорі є окремим випадком, що не розглядається у рамках даного дисертаційного дослідження. Тому, при розрахунку сумарної потужності споживання обчислювального вузла програмним методом із деяким наближенням можливо вважати, що потужність споживання вузла є сумою потужностей споживання його компонентів при ступені завантаженості рівним ступеню завантаженості ЦП.

У результаті отримуємо функцію $P_j = f(CPU_j)$ як залежність потужності споживання j-го обчислювального вузла від ступеня завантаженості ЦП у табличному вигляді, так само як і у випадку використання емпіричного методу. Для отримання аналітичного виду функції так само можуть бути використані методи апроксимації або інтерполяції.

Програмний метод визначення моделей енергоспоживання володіє дещо нижчою точністю у порівнянні з емпіричним визначенням моделей. Однак, його простота дозволяє проводити розрахунок “на льоту” без переривання роботи системи при введені у систему нових обчислювальних вузлів. У рамках даного дисертаційного дослідження особливо важливими є відносні значення потужностей споживання обчислювальних вузлів, а не абсолютні значення потужностей споживання, що виступає додатковим аргументом у разі використання програмного методу.

Таким чином, з урахуванням переваг та недоліків розглянутих методів визначення моделей енергоспоживання обчислювальних вузлів, емпіричний метод пропонується використовувати при початковому налаштуванні системи для більш точного визначення залежностей потужності споживання вузлів від ступеня їх завантаженості. У випадку подальшої реконфігурації системи шляхом введення додаткових обчислювальних вузлів або проведення її періодичної переатестації пропонується використання програмного методу, як більш швидкого та такого, що не перериває функціонування системи. В обох випадках для отримання аналітичних функцій пропонується використання методу інтерполяції поліномом ступеня n, основною перевагою якого є точне представлення значень функції енергоспоживання у опорних точках (вузлах інтерполяції) та достатня точність представлення функції на проміжках між вузлами інтерполяції.

ВИСНОВКИ

1. Проведено формалізацію процесу обслуговування навантаження в ІКМ за допомогою онтологічної моделі. Це дозволило систематизувати та якісно описати складні взаємозв'язки між виділеними показниками ефективності досліджуваного процесу та параметрами, що впливають на них.

2. Запропоновано математичну модель системи розподіленого обслуговування навантаження ІКМ у вигляді СМО, що відрізняється від відомих тим, що враховує змінний характер інтенсивності вхідного навантаження та можливість використання технік паралелізації при розробці ПЗ сучасних ІКМ, що є науковою новизною. Це дозволило отримати кількісний опис взаємозв'язків між параметрами системи обслуговування та показниками ефективності процесу обслуговування навантаження ІКМ.

3. В процесі математичного моделювання системи розподіленого обслуговування навантаження як СМО запропоновано метод переходу від нестаціонарного неординарного вхідного потоку заявок до стаціонарного ординарного потоку шляхом дискретизації кривої інтенсивності вхідного навантаження та за допомогою переходу до комплектів серверів. Це дозволило значно спростити розрахунки. Можливі втрати точності моделі в процесі спрощення регулюються параметрами трансформованої моделі, що дозволяє зберегти адекватність моделі об'єктові дослідження.

4. Для дискретизації кривої інтенсивності вхідного навантаження запропоновано використання методу квантування за рівнями, що дозволило узгодити величину кроку дискретизації функції зі швидкістю зміни інтенсивності вхідного навантаження.

5. Для визначення кроку квантування запропоновано метод розрахунку порогових величин інтенсивностей вхідного навантаження $\lambda(t)_{\text{п}}$ з урахуванням кількості обчислювальних вузлів у системі.

6. У результаті математичного моделювання отримано математичні вирази для показників ефективності СМО: імовірності втрати запиту у змодельованій

системі, її абсолютної пропускної здатності, середньої кількості заявок у черзі та в обробці.

7. Описано два методи отримання математичної моделі енергоспоживання обчислювальних вузлів розподілених ЦОД $P_j = f(CPU_j)$: аналітичний та емпіричний. Емпіричний метод є більш складним, у той час як аналітичний метод є простим та може використовуватись без переривання роботи системи при введені до неї нових вузлів або під час її переатестації. Недоліком аналітичного методу є менша точність отриманих моделей енергоспоживання, однак, обидва методи потребують більш детального порівняння ефективності шляхом їх експериментального аналізу.

8. З огляду на переваги та недоліки проаналізованих методів, рекомендовано використання методу емпіричного визначення математичних моделей енергоспоживання для нових систем на етапі їх налаштування. У випадку введення нових вузлів до системи або під час її переатестації, рекомендовано використання аналітичного методу визначення моделей енергоспоживання.

9. Для обох методів запропоновано використання математичного методу інтерполяції функцій поліномом ступеня n з метою перетворення функцій з табличного в аналітичний вигляд.

РОЗДІЛ 3
КОМПЛЕКСНИЙ МЕТОД ЕНЕРГОЕФЕКТИВНОГО ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ МЕРЕЖІ

У розділі наведено опис запропонованого комплексного методу енергоефективного обслуговування навантаження інформаційно-комунікаційної мережі. Опис запропонованого комплексного методу та окремих його кроків опубліковано у роботах [36 - 39, 52].

3.1 Комплексний метод енергоефективного обслуговування навантаження

Запропонований комплексний метод енергоефективного обслуговування навантаження складається з 4 основних кроків, що виконуються у дві фази (Рисунок 3.1). При розробці комплексного методу були взяті до уваги загальні рекомендації щодо підвищення енергоефективності обробки навантаження ІКМ ITU-T L.1300 [96].

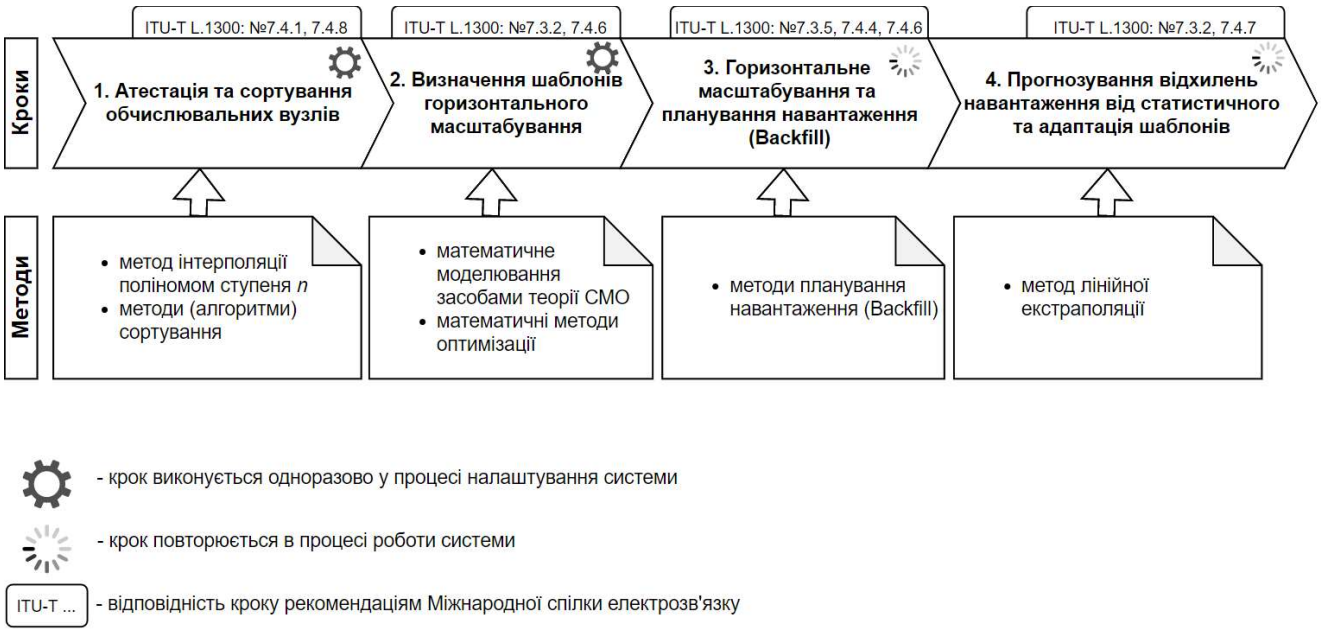


Рисунок 3.1 Спрощена схема, що відображає кроки запропонованого комплексного методу та окремі використані методи

<u>Фаза налаштування:</u>

***Крок 1*:** Атестація та сортування обчислювальних вузлів. На цьому кроці вирішується питання “Які обчислювальні вузли слід використовувати більш інтенсивно?”. Метою даного кроку є оцінка потужності споживання $P_j(CPU_j)$ та продуктивності C_j кожного з обчислювальних вузлів та ранжування вузлів за цими показниками. На даному кроці використовуються методи інтерполяції поліномом ступеня n та сортування. Цей крок відповідає рекомендаціям 7.4.1 та 7.4.8 Міжнародної спілки електрозв’язку ITU-T L.1300 [96], а саме:

– 7.4.1 Аудит (атестація) наявного фізичного обладнання та послуг;

– 7.4.8 Перевірка наявного інформаційно-комунікаційного обладнання щодо відповідності вимогам щодо захисту довкілля.

***Крок 2*:** Визначення шаблонів горизонтального масштабування. Метою цього кроку є відповідь на питання “Скільки активних вузлів потрібно для обслуговування статистичного навантаження у кожен момент часу для задоволення критерія ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$?”. На цьому кроці проводиться аналіз статистистичної кривої навантаження у системі з використанням побудованої математичної моделі системи у вигляді СМО та математичної оптимізації. Цей крок відповідає рекомендаціям 7.3.2 та 7.4.6 Міжнародної спілки електрозв’язку ITU-T L.1300 [96], а саме:

– 7.3.2 Зменшення рівня доступності апратного забезпечення, якщо це можливо (тобто, для кожної послуги має бути визначений той рівень доступності АЗ, який фактично виправданий, і саме цього рівня варто дотримуватись);

– 7.4.6 Консолідація наявних сервісів.

<u>Оперативна фаза:</u>

***Крок 3*:** Горизонтальне масштабування та планування навантаження. На цьому кроці вирішується питання “Як розподілити навантаження між активними на даний момент обчислювальними вузлами?”. Метою цього кроку є забезпечення

максимально можливого завантаження активних ресурсів CPU_j у системі, мінімізація простою обладнання та, як результат, підвищення продуктивності системи C_Σ шляхом збільшення обсягу обчислювального навантаження ω, що обробляється системою за час T: $C_\Sigma = \frac{\omega}{T}$. При цьому, обчислювальні ресурси, що не використовуються, виводяться із активного режиму з метою зменшення загального енергоспоживання усіх N вузлів $\sum_{j=1}^{N} W_j$. На цьому кроці використовується модифікований метод горизонтального масштабування системи у комбінації з енергоефективним плануванням навантаження. Крок 3 відповідає рекомендаціям 7.3.5, 7.4.4 та 7.4.6 Міжнародної спілки електрозв'язку ITU-T L.1300 [96], а саме:

– 7.3.5 Обрання / розробка ефективного програмного забезпечення;

– 7.4.4 Вимкнення обладнання, що знаходиться у стані простою;

– 7.4.6 Консолідація наявних сервісів.

***Крок 4*:** Прогнозування відхилень навантаження від статистичного та адаптація шаблонів. Питання, що вирішується на цьому кроці: "Як впоратися з динамічними відхиленнями навантаження від статистичного у режимі реального часу?". На цьому кроці проводиться аналіз інтенсивності поточного навантаження $\lambda_{obs}(t)$ та її порівняння із статистичною інтенсивність навантаження для даного моменту часу $\lambda_{pred}(t)$. У разі виявлення відхилень проводиться реконфігурація системи для задоволення обраного критерія ефективності $K_{opt} = E_\Sigma * C_\Sigma \rightarrow max,$ при $p_{avail} \geq p_{avai\ SLA}$. Окрім цього здійснюється прогнозування інтенсивності навантаження на наступний інтервал часу, довжина якого залежить від ступеня вивленого відхилення. На цьому кроці використовується метод лінійної екстраполяції. Крок 4 відповідає рекомендації 7.3.2 та 7.4.7 Міжнародної спілки електрозв'язку ITU-T L.1300 [96], а саме:

– 7.3.2 Зменшення рівня доступності апратного забезпечення, якщо це можливо (тобто, для кожної послуги має бути визначений той рівень доступності АЗ, який фактично виправданий, і саме цього рівня варто дотримуватись);

– 7.4.7 Контроль за енергоспоживанням системи.

Фаза налаштування виконується один раз для незмінної конфігурації системи та статистики навантаження. Етапи, що належать до оперативної фази, виконуються безперервно під час роботи системи.

3.1.1 Атестація та сортування обчислювальних вузлів

Першим кроком у запропонованому комплексному методі є атестація та сортування обчислювальних вузлів.

Цей крок в свою чергу включає в себе такі процеси:

1. Визначення залежностей енергоспоживання обчислювальних вузлів від ступеня завантаженості їх ЦП CPU_j у табличному вигляді;

2. Представлення залежності енергоспоживання кожного j-го обчислювального вузла в аналітичному вигляді $P_j(CPU_j)$;

3. Сортування вузлів за показниками продуктивності та енергоспоживання.

Процеси 1 - 2 об’єднані загальним терміном “атестація обчислювальних вузлів”. Метою процесу атестації є визначення характеристики енергоспоживання кожного j-го обчислювального вузла при різних рівнях завантаженості ЦП CPU_j у вигляді функції $P_j = f(CPU_j)$ в аналітичному вигляді; окрім характеристики енергоспоживання у процесі атестації береться до уваги також показник продуктивності обчислювального вузла.

Визначення залежностей енергоспоживання обчислювальних вузлів у табличному вигляді проводиться одним із способів:

– ***емпіричним*** – шляхом безпосереднього вимірювання потужності споживання у деяких опорних точках з подальшим отриманням аналітичної функції шляхом інтерполяції отриманих даних поліномом ступеня n;

– ***програмним*** – шляхом програмного визначення номінальних потужностей споживання окремих компонентів обчислювального вузла та подальшого отриманням аналітичної функції шляхом інтерполяції отриманих даних поліномом ступеня n.

У випадку використання емпіричного способу, на кожен з обчислювальних вузлів системи встановлюється спеціалізоване програмне забезпечення (наприклад, утиліта $stress - ng$), що здатне працювати як стресс-тест для ЦП вузла повністю завантажуючи вказану користувачем кількість обчислювальних ядер на заданий період часу. За допомогою цього програмного забезпечення на кожному вузлі почергово штучно створюється навантаження CPU_j бажаної величини (наприклад, 25, 50, 75, 100%). При кожному значенні навантаження здійснюється вимірювання середніх на інтервалі часу миттєвих значень напруги та струму безпосередньо у колі живлення кожного сервера. На основі отриманих значень напруги та струму з урахуванням похибок розраховується середнє миттєве значення потужності за формулою:

$$P(CPU_j) = \bar{U}(CPU_j) * \bar{I}(CPU_j), \qquad (3.1)$$

де $\bar{U}$ та $\bar{I}$– визначені значення напруги та струму відповідно; CPU_j – заданий рівень завантаженості ЦП обчислювального вузла. Аналогічні дії проводяться для всіх N обчислювальних вузлів досліджуваної системи при всіх M ступенях навантаження ЦП. Отримана у результаті матриця розміром M x N містить у стовпцях функції енергоспоживання вузлів системи у табличному вигляді (2.22).

У випадку використання програмного способу визначення залежностей енергоспоживання обчислювальних вузлів у табличному вигляді, використовуються номінальні значення потужностей споживання складових компонентів обчислювального вузла (ЦП, графічного процесора, елементів оперативної пам’яті та материнської плати, елементів постійної пам’яті, дисплея (за наявності)). Для отримання цих значень можуть бути використані дані вимірювань з відкритих Інтернет-джерел, наприклад SPEC (Standard Performance Evaluation Corporation) [124].

Сумарна потужність споживання обчислювального вузла при кожному розглядуваному рівні завантаженості процесора CPU_j від 0 до 100% в такому випадку розраховується за формулами (2.33). Отримані значення

$P_{\Sigma_j}(CPU_j = 0), \ldots P_{\Sigma_j}(CPU_j = 100)$ являють собою функцію енергоспоживання кожного вузла у табличному вигляді (так само як при використанні емпіричного методу).

Наступним кроком є ***представлення залежностей енергоспоживання в аналітичному вигляді*** методом інтерполяції поліномом ступеня $n = m - 1$. У результаті процесу інтерполяції, залежність потужності споживання від завантаженості обчислювального вузла представляється поліномом вигляду:

$$P_j(CPU_j) = a_0 + a_1 * CPU_j^1 + a_2 * CPU_j^2 + \ldots + a_{m-1} * CPU_j^{m-1}, \qquad (3.2)$$

де m – кількість вузлів інтерполяції, що відповідає кількості вимірювань енергоспоживання при сталій завантаженості процесора; CPU_j – завантаженість ЦП обчислювального вузла; $a_0, a_1, \ldots a_{m-1}$ – коефіцієнти полінома.

Сортування вузлів за показниками продуктивності та енергоспоживання проводиться з метою отримання ранжованого списку обчислювальних вузлів з метою більш інтенсивного подальшого використання “найкращих” вузлів у списку. Сортування проводиться за таким алгоритмом:

1. Розрахунок показника енергоспоживання вузлів S_j як інтеграла під кривою енергоспоживання для кожного вузла за формулою:

$$S_j = \int_0^{100} P_j(CPU_j) dCPU_j. \qquad (3.3)$$

Інтеграл під кривою енергоспоживання дозволяє порівнювати потужності споживання вузлів при різних рівнях завантаженості їх процесорів.

2. Сортування масиву обчислювальних вузлів за значенням показника енергоспоживання S_j (3.3) у порядку спадання. В результаті отримаємо масив $N_S = \{N_{S_{j_{max}}}, \ldots N_{S_j min}\}$.

3. Сортування масиву обчислювальних вузлів за значенням показника їх номінальної продуктивності C_j у порядку зростання (нагадаємо, що значення номінальної продуктивності кожного j-го вузла C_j є однією із заданих характеристик обчислювальної системи). В результаті отримаємо масив

$N_{\mathrm{C}} = \{N_{\mathrm{C}_{j_{min}}}, \dots N_{\mathrm{C}_{jmax}}\}$.

В залежності від позиції вузла із номером j у сортованих масивах N_S та N_C, він отримує два вагових коефіцієнта (дві оцінки): $mark_S$ і $mark_C$ відповідно. При чому $mark_S, mark_C = \overline{1, N}$, де N – загальна кількість вузлів у системі. $mark_S$ рівна позиції *j-го* вузла у сортованому масиві N_S, $mark_C$ – позиції цього вузла у сортованому масиві N_C відповідно. Тоді, сумарна оцінка кожного вузла визначається за рівністю: $mark_{\Sigma_j} = mark_{C_j} + mark_{S_j}$.

Таким чином, вузли можуть бути проранжовані від найгіршого до найкращого відповідно до оцінок $mark_{\Sigma_j}$. У результаті отримуємо список вузлів N_{mark_Σ} впорядкований згідно з процедурою ранжування.

3.1.2 Метод визначення шаблонів горизонтального масштабування

Як вже згадувалось, метою технік горизонтального масштабування є адаптація обсягу активних обчислювальних ресурсів під обсяги вхідного навантаження, що потребує обробки. Одним з можливих підходів при цьому є переведення частини обчислювальних вузлів у режим сну. Тому, при використанні горизонтального масштабування постає питання визначення оптимальної кількості активних обчислювальних вузлів у системі у кожен момент часу. На прийняття рішення про оптимальну кількість вузлів у системі впливають визначені показники ефективності: енергоефективності, продуктивності та доступності системи. Очевидно, що зменшення кількості активних вузлів призводить до зниження сумарного енергоспоживання системи, що може позитивно вплинути на показник енергоефективності. Водночас зменшення обсягу активних ресурсів може призвести до зниження продуктивності системи та до зниження її доступності у випадку, коли нестача обчислювальних ресурсів призводить до втрати обчислювальних робіт.

Таким чином, виникає задача побудови так званих шаблонів горизонтального масштабування, що визначають оптимальну кількість вузлів за показниками енергоефективності, продуктивності та доступності системи для кожного моменту

часу. Звичайно, у загальному випадку інтенсивність вхідного навантаження для розподілених ЦОД інформаційно-комунікаційної мережі є випадковою величиною, що з одного боку робить неможливим визначення оптимальної кількості обчислювальних вузлів наперед. Однак, при тривалому спостереженні можливо помітити, що характер зміни інтенсивності вхідного навантаження кожного окремого типу (SDN, NFV, Network Slicing, Edge Computing, bDDN) за деякий період часу (доба, тиждень, місяць) є сталим та може бути виражений за допомогою статистичної кривої навантаження за відповідний період часу. Це дозволяє визначати шаблони горизонтального масштабування на основі статистичних даних про інтенсивність вхідного навантаження.

Для визначення шаблонів масштабування скористаємося побудованою у підрозділі 2.2 математичною моделлю системи розподіленого обслуговування навантаження у вигляді СМО. Побудована модель дозволила визначити показник доступності системи, виражений у вигляді імовірності втрати запиту, як функцію кількості серверів СМО k':

$$p_{loss} = p_{k'+Q} = \frac{\rho^{k'+Q}}{k'^{Q} * k'!} p_0, \tag{3.4}$$

де $\rho = \lambda'/\mu'$; $p_0 = (1 + \frac{\rho}{1!} + \frac{\rho^2}{2!} + \cdots + \frac{\rho^{k'}}{k'!} + \frac{\frac{\rho^{k'+1}}{k'k'!} * \left(1 - \left(\frac{\rho}{k'}\right)^Q\right)}{1 - \frac{\rho}{k'}})^{-1}$.

У побудованій моделі сервер являє собою обчислювальний вузол із середньою інтенсивністю обробки $\mu' = \frac{\sum_{j=1}^{k'} \sum_{i=1}^{M(x)} \mu_{ij}}{k'}$ робіт в одиницю часу, який має $M(x)$ обчислювальних ядер. При досить великій кількості серверів за законом великих чисел [112] середнє значення їх інтенсивності обробки та кількості ядер наближаються до математичного очікування μ' та $M(x)$ відповідно. Тому, у формулі (3.4) можна вважати параметр k' (кількість серверів СМО) еквівалентним параметру n (кількість активних обчислювальних вузлів): $k' \sim n, n \gg 1$.

Тому, як видно з виразу (3.4), на імовірність втрати запиту впливає кількість обчислювальних вузлів n, інтенсивності надходження та обробки запитів (λ' та μ'),

а також довжина черги у системі Q. У рамках підходу горизонтального масштабування змінним параметром виступає кількість активних обчислювальних вузлів. Тобто, вираз (3.4) дозволяє визначити вплив горизонтального масштабування на імовірність втрати запиту у системі p_{loss} як характеристику доступності системи.

У рамках кроку 1 (зокрема атестації обчислювальних вузлів) визначено функції енергоспоживання обчислювальних вузлів в аналітичному вигляді. Ці функції мають вигляд поліномів (3.2). Тоді, залежність сумарної потужності споживання усієї системи від завантаженості окремих серверів визначається сумою функцій енергоспоживання усіх активних серверів:

$$P_{\Sigma} = \sum_{j=1}^{n} P_j\left(CPU_j\right), \tag{3.5}$$

де CPU_j – завантаженість процесора $j-$го вузла; n – кількість активних обчислювальних вузлів у системі.

Практичне використання виразу (3.5) для підрахунку сумарного енергоспоживання системи ускладнено тим фактом, що завантаженість процесора кожного $j-$го вузла CPU_j – це випадкова величина із деяким середнім значенням CPU_{avg_j}. Тобто, неможливо визначити сумарну потужність споживання системи у певний момент часу, оскільки невідомою є поточна завантаженість кожного з обчислювальних вузлів. Крива вхідного навантаження, що розглядається у якості вхідних даних системи, не дає інформації про завантаженість окремих вузлів системи, оскільки їх завантаженість регулюється планувальником навантаження та фактично не залежить від загальної інтенсивності вхідного навантаження.

В залежності від використовуваного планувальника, середня завантаженість вузлів CPU_{avg_j}може різнитися. Так, наприклад, ряд планувальників, що в основі використовують підхід Backfill [29] націлені на підтримання постійно високого рівня завантаженості вузлів системи з метою якомога ефективнішого застосування

АЗ. У цьому випадку, середня завантаженість вузлів системи зазвичай сягає 75% [125], а відхилення від середньої завантаженості є незначним.

Оскільки у роботі пропонується використовувати планувальник навантаження, що базується на стратегії Backfill, то при подальших розрахунках енергоспоживання системи коректно використовувати середній рівень завантаженості процесора CPU_{avg_j}. Для інших планувальників у випадку, наприклад, якщо завантаженість кожного з вузлів розподілена за рівномірним законом, таке припущення не є коректним.

Таким чином, сумарну потужність споживання вузлів системи у випадку використання стратегії планування Backfill коректніше розраховувати за формулою:

$$P_{\Sigma} = \sum_{j=1}^{n} P_j \left(CPU_{avg_j} \right). \tag{3.6}$$

Як видно із формули (3.6), сумарна потужність споживання системи також є функцією кількості активних вузлів в системі n.

Визначивши залежності імовірності втрати запиту p_{loss} та сумарної потужності споживання P_{Σ} від кількості активних вузлів системи n, сформулюємо задачу оптимізації: знайти таке значення n, при якому цільова функція:

$$P_{\Sigma} = \sum_{j=1}^{n} P_j \left(CPU_{avg_j} \right) \rightarrow min \tag{3.7}$$

набуватиме мінімального значення при обмеженні:

$$p_{loss} = \frac{\rho^{n+Q}}{n^{Q} * n\,!} p_0 = \frac{(\frac{\lambda'}{\mu'})^{n+Q}}{n^{Q} * n\,!} p_0 < p_{loss_{SLA}}, \tag{3.8}$$

де $n \gg 1, n \in \mathbb{Z}$ – кількість активних обчислювальних вузлів кластера ($\mathbb{Z}$ – множина цілих чисел); Q – довжина черги в системі; λ' – інтенсивність надходження запитів, μ' – інтенсивність обробки запитів; $p_{loss_{SLA}}$ – визначена згідно з SLA максимально допустима імовірність втрати заявки користувача.

В загальному випадку отримана оптимізаційна задача (3.7) - (3.8) має своє оптимальне рішення для кожного моменту часу (для кожного значення змінної інтенсивності надходження запитів λ'). Однак, як описувалось у підпункті 2.2.2.2 , можливим є визначення інтервалів часу Δt, на яких інтенсивність надходження запитів λ^t можна вважати сталою. Для цього скористаємось описаною у підпункті 2.2.2.2 ідеєю квантування та визначимо порогові значення інтенсивностей надходження запитів $\lambda(t)_{\text{п}}$ за допомогою виразу:

$$\lambda(t)_{\text{п}} = \mu' * \sqrt[n+Q]{\frac{p_{loss_{SLA}} * n^Q * n!}{p_0}}, \qquad n \in \mathbb{N}^* \tag{3.9}$$

при заданій допустимій імовірності втрати запиту $p_{loss_{SLA}}$, де $n \in \mathbb{N}^*$ – кількість активних серверів у системі, що є натуральним числом від 1 до N; Q – довжина черги; μ' – середня інтенсивність обробки обчислювального вузла.

Шляхом послідовних розрахунків виразу (3.9) для $n \in \mathbb{N}^*, n = (\overline{1, N})$ отримуємо послідовність порогових значень $\lambda(t)_{\text{п}}$ при досягненні яких виникає необхідність зміни кількості активних обчислювальних вузлів у системі.

При цьому, оскільки залежність сумарної потужності споживання системи від кількості активних вузлів (3.7) є монотонно зростаючою функцією, підтримки мінімально необхідної кількості вузлів для забезпечення виконання умови (3.8) є достатньою для одночасної мінімізації функції (3.7). Тобто, достатнім для ***визначення шаблонів масштабування*** є застосування такого ***методу***:

1. При заданій максимально допустимій імовірності втрати запиту $p_{loss_{SLA}}$ визначити порогові значення інтенсивності вхідного навантаження згідно виразу (3.9).

2. Для добової статистики навантаження для кожного з інтервалів Δt, що відповідають переходу від значення $\lambda(t)_{\text{п},i}$ до $\lambda(t)_{\text{п},i+1}$, визначити $\lambda' = max(\lambda_i), \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$.

3. Для кожного з інтервалів Δt визначити мінімально необхідну кількість обчислювальних вузлів n, при якій нерівність (3.8) буде справджуватись при

$\lambda' = max(\lambda_i), \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$.

Приклад результату визначення шаблонів масштабування наведено на рисунку (Рисунок 3.2).

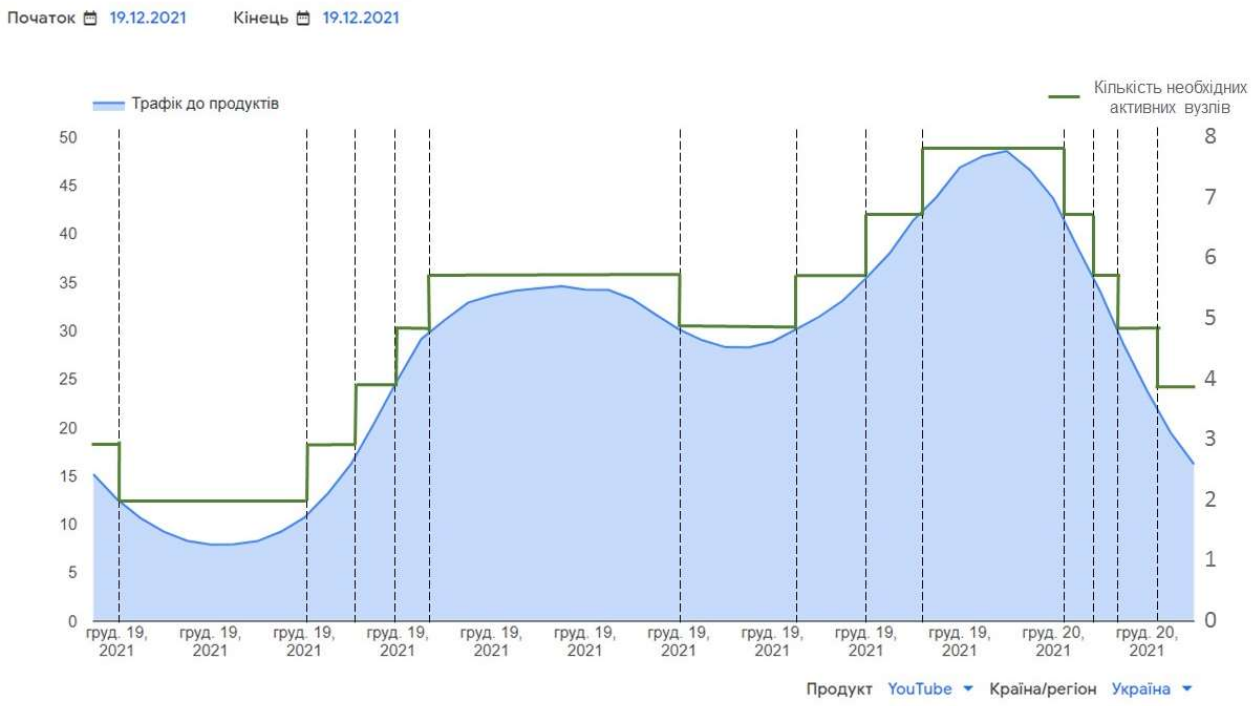


Рисунок 3.2 Приклад результату визначення шаблонів масштабування для статистичної кривої навантаження (на прикладі статистики навантаження на відео-сервіс) [111]

Ці шаблони пропонується створювати статично, тобто один раз для визначеної статистичної кривої навантаження. Оскільки шаблони створюються статично, затрати ресурсів (як часових так і обчислювальних) на цей процес є мінімальними.

Задля гарантування доступності системи у моменти переходу від одного шаблону до іншого, варто ввести деяку надлишковість, шляхом переведення додаткового вузла (або декількох вузлах, в залежності від масштабів системи) в «активний стан». Це також дозволить системі успішно обробляти навантаження, інтенсивність якого дещо відхиляється від статистичного. При цьому розрахунок кількості вузлів слід проводити для значень $max(\lambda_i) + \sigma, \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$ де σ – стандартне відхилення визначене для кривої добового навантаження.

Реальне навантаження в системі може значно різнитись від статистичного (наприклад у період свят або проведення масових заходів), тому, при виявленні відхилення від статистичного навантаження, що перевищує допустимі норми, пропонується застосування динамічного коригування шаблонів масштабування системи.

Таким чином, у рамках фази налаштування в результаті проведення попередньої атестації обчислювальної системи визначено індивідуальні залежності енергоспоживання вузлів системи. На основі отриманих залежностей та даних про продуктивність кожного з вузлів, вузли системи проранжовано від найкращого до найгіршого за сукупним показником енергоефективності та продуктивності. На основі статистичної кривої інтенсивності вхідного навантаження побудовано шаблони горизонтального масштабування системи.

3.1.3 Застосування визначених шаблонів для горизонтального масштабування та енергоефективного планування задач

На відміну від попередніх, цей крок належить до оперативної фази та повинен періодично виконуватись у процесі обслуговування навантаження ІКМ. Він має на увазі застосування удосконаленого підходу горизонтального масштабування та енергоефективного планування навантаження.

Застосування удосконаленого підходу горизонтального масштабування

У результаті кроків 1 та 2 визначено, які вузли системи є найбільш ефективними за показниками енергоефективності та продуктивності, а також сформовано шаблони масштабування системи, що вказують, скільки вузлів повинно використовуватись у кожен момент часу задля забезпечення доступності системи та мінімізації її енергоспоживання. Результати цих кроків застосовуються у рамках удосконаленого методу горизонтального масштабування: згідно з визначеними шаблонами у процесі роботи системи частина вузлів переводиться у режим сну, при чому переведенню в режим сну підлягають ті вузли, які знаходяться нижче у ранжованому списку вузлів від найбільш до найменш ефективного. Це

дозволяє підтримувати у системі лише мінімально необхідну кількість найбільш ефективних активних обчислювальних ресурсів у кожен момент часу.

<u>Застосування енергоефективного планування навантаження</u>

На основі аналізу залежностей потужності споживання від завантаженості вузла (Рисунок 1.4), а також експериментально отриманих у роботі функцій $P_j = f(CPU_j)$ (підпункт 4.3.1.2), можна зробити висновок, що приріст енергоспоживання обчислювальних вузлів тим нижчий, чим вищим є абсолютне значення попередньої завантаженості (тобто, крива має характер близький до логарифмічного). З цього випливає, що більшої енергоефективності можна досягти при максимальній середній завантаженості вузлів, що одночасно збільшить продуктивність обробки (адже середня завантаженість процесора вузла тим більша, чим більше задач він обробляє в середньому за одиницю часу). Тобто, задля збільшення енергоефективності та продуктивності системи необхідно завантажувати обчислювальні вузли максимально у кожен момент часу t.

Для досягнення цієї мети у якості планувальника задач у системі пропонується використовувати планувальник на основі стратегії Backfill [29].

Розглянемо алгоритм, за яким відбувається розподіл задач у системі відповідно до стратегії Backfill.

Нехай у момент часу $t-1$ у системі немає вільних ресурсів для виконання жодної задачі із черги. У момент часу t одна із задач завершує обробку – звільнюється відповідна кількість ресурсів на $j-$му вузлі. У цей момент планувальник Backfill здійснює наступні кроки:

Крок 1. Визначення обсягу доступних ресурсів $\left\{RAM_{avail_j}, cores_{avail_j}, storage_{avail_j}\right\}$ на кожному обчислювальному вузлі.

Крок 2. Якщо в системі достатньо ресурсів для виконання першої роботи з черги $\{RAM_{min_1}, cores_{min_1}, storage_{min_1}\}$ – перша робота з черги надходить на обробку. Перехід на Крок 1.

Інакше – обчислюється момент часу, коли ресурсів стане достатньо для виконання першої роботи з черги (на основі значення Δt_{max_1}), і проводиться резервування ресурсів для даної роботи. Перехід на Крок 3.

Крок 3. Перехід до наступної задачі в черзі. Перевірка умов:

– Чи достатньо ресурсів для виконання другої роботи з черги $\{RAM_{min_2}, cores_{min_2}, storage_{min_2}\}$?

– Чи не порушує друга задача резервування, що здійснено на кроці 2 ($\Delta t_{max_1} > \Delta t_{max_2}$)?

Якщо обидві вимоги виконуються – друга задача надходить на обробку. Перехід на Крок 1.

Інакше – Крок 3 повторюється для наступної задачі у черзі.

Таким чином забезпечується позачергове виконання задач, що не вимагають значних ресурсів задля уникнення простою обчислювальних вузлів. Одночасно, для задач, що вимагають великого обсягу ресурсів, але згідно черги мають бути виконані раніше, гарантується, що вони будуть виконані одразу ж після звільнення достатньої кількості ресурсів для них.

Псевдокод алгоритму наведено на рисунку (Рисунок 3.3).

```
1  for each j-th node {
2      for each job {
3          avail_resources = node[j].resources;
4          if (job.resources <= avail.resources){
5              node[j].process(job);
6              break;
7          }
8          else{
9              node[j].reserve(job.resources) after t_current_processing;
10             if(job.next.resources <= avail.resources &&
11                job.next.processing_time < t_current_processing){
12                 node[j].process(job.next);
13                 break;
14             }
15             else{
16                 continue;
17             }
18         }
19     }
20 }
```

Рисунок 3.3 Псевдокод алгоритму планування навантаження на основі стратегії Backfill

Використання стратегії Backfill дозволяє ефективно використовувати обчислювальні вузли, додатково збільшуючи енергоефективність та продуктивність розподіленої обчислювальної системи.

3.1.4 Виявлення відхилення поточного навантаження від статистичного та динамічна адаптація шаблонів масштабування

У пунктах 3.1.2 та 3.1.3 описано методи підвищення енергоефективності та продуктивності обчислень, що базуються на статистичних даних про добове вхідне навантаження на систему. Однак, унаслідок певних подій (святкові дні, проведення масових заходів, тощо), навантаження на розподілену систему в цілому та серверний кластер зокрема може мати значне відхилення від статистичного. У такому випадку пропонується застосувати модифікований підхід щодо динамічного керування ресурсами, запропонований авторами роботи [126] для віртуалізованих мережевих функцій. Основна ідея підходу – динамічна зміна інтенсивності моніторингу поточного навантаження та прогнозування навантаження на наступний часовий період на основі аналізу відхилення поточного реального навантаження від статистичного. Запропонована модифікація підходу описана у роботах [44, 45].

Згідно з запропонованою у пункті 2.1.2 онтологічною моделлю досліджуваної системи, складовою частиною ПЗ розподіленої обробки даних є блок моніторингу та прогнозування навантаження. Задача блоку моніторингу – здійснювати періодичні виміри інтенсивності вхідного навантаження. Нехай визначено деякий базовий період моніторингу I_{base}. Цей період може обраний довільно і коригуватись на основі даних про відхилення навантаження, однак, при виборі I_{base} необхідно враховувати той факт, що при високій частоті моніторингу зростає обсяг додаткових накладних ресурсів, необхідних для його здійснення.

Нехай у деякий момент часу t поточна інтенсивність вхідного навантаження $\lambda_{obs}(t)$ перевищила з урахуванням допустимого відхилення значення статистичного навантаження $\lambda_{pred}(t)$, прогнозованого на цей момент часу. Ця

подія є сигналом до зміни частоти моніторингу I_{base}. Змінювати частоту моніторингу пропонується за формулою, отриманою на основі дослідження [126]:

$$W(t) = I_{base} - K * \frac{\sum_{j=t-h}^{t-1} max(\sigma; |\lambda_{obs}(j) - \lambda_{pred}(j)|)}{h-1} I_{base},$$
$$K = \begin{cases} 0, \sigma > |\lambda_{obs}(j) - \lambda_{pred}(j)| \\ const, \sigma \leq |\lambda_{obs}(j) - \lambda_{pred}(j)| \end{cases}, \quad (3.10)$$

де I_{base} – базовий період моніторингу;

σ – допустиме відхилення інтенсивності вхідного навантаження;

$\lambda_{obs}(j)$ – інтенсивність навантаження, що спостерігалась у момент часу t;

$\lambda_{pred}(j)$ – очікувана (прогнозована) інтенсивність навантаження у момент часу t, згідно з кривою статистики навантаження;

h – кількість попередніх інтервалів моніторингу, визначена алгоритмом (ця величина може бути обрана в залежності від особливостей системи, зокрема, можливостей щодо зберігання результатів моніторингу);

K – константа нормалізації, що визначається для кожної системи індивідуально та забезпечує невід'ємність виразу (3.10). K повинно бути визначеним таким чином, що при будь-яких реально можливих відхиленнях $\lambda_{obs}(t)$ від $\lambda_{pred}(t)$ виконувалась нерівність $\frac{\sum_{j=t-h}^{t-1} max(\sigma; |\lambda_{obs}(j) - \lambda_{pred}(j)|)}{h-1} < \frac{1}{K}$.

Формула (3.10) визначає величину так званого вікна адаптації, для якого повинна бути розрахована нова прогнозована інтенсивність вхідного навантаження.

Як видно з формули (3.10), при відсутності відхилення поточного навантаження від статистичного, розмір вікна адаптації є рівним I_{base} (тобто моніторинг навантаження здійснюється з нормальною частотою). Однак, при значенні відхилення $\lambda_{obs}(t)$ від $\lambda_{pred}(t)$, що перевищує значення допустимого відхилення σ, інтервал моніторингу коригується на величину пропорційну $|\lambda_{obs}(j) - \lambda_{pred}(j)|$.

Припустимо, що у момент часу t було зафіксовано перевищення відхилення $\lambda_{obs}(t)$ від $\lambda_{pred}(t)$ та розраховано величину вікна адаптації (3.10). Це не тільки

означає, що наступного разу моніторинг поточної інтенсивності навантаження буде здійснено у момент часу $t + W(t)$, але й показує необхідність реагування на перевищення допустимого відхилення інтенсивності вхідного навантаження. Це реагування пропонується здійснювати шляхом адаптації побудованих на кроці 2 (пункт 3.1.2) шаблонів горизонтального масштабування. Адаптація шаблонів пропонується здійснювати за допомогою використання методу лінійної екстраполяції. Для цього здійснюється прогнозування максимуму інтенсивності навантаження на наступний період, що відповідає шаблону масштабування. Прогнозування, згідно з роботою [126], пропонується здійснювати за формулою:

$$\lambda_{pred}(t + \Delta t) = \lambda_{basepred}(t + \Delta t) + \sum_{j=t-h}^{t-1} \frac{\lambda_{obs}(j) - \lambda_{pred}(j)}{h - 1}, \qquad (3.11)$$

де $\lambda_{basepred}(t + \Delta t) = max(\lambda_i), \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$ – максимальна величина інтенсивності вхідного навантаження згідно статистики на період, для якого має бути скоригований шаблон масштабування (між сусідніми пороговими значеннями $\lambda(t)_{\text{п}}$ відповідно);

$\sum_{j=t-h}^{t-1} \frac{\lambda_{obs}(j) - \lambda_{pred}(j)}{h-1}$ - середнє відхилення навантаження, що спостерігалось протягом h попередніх інтервалів моніторингу.

Точки $\lambda_{pred}(t + \Delta t)$ та $\lambda_{obs}(t)$, для якої спостерігалось перевищення допустимого відхилення, формують пряму, що відображає результат лінійної екстраполяції. Після чого проводиться перерахунок шаблону масштабування між сусідніми пороговими величинами $\lambda(t)_{\text{п},i}$ та $\lambda(t)_{\text{п},i+1}$ згідно із формулою (3.8).

Принцип роботи методу адаптації шаблонів масштабування зображено на рисунку (Рисунок 3.4).

Застосування цього підходу дозволить прогнозувати інтенсивність вхідного навантаження на наступні періоди часу на основі спостереження за відхиленням навантаження від свого статистичного значення у попередні періоди часу. На основі спрогнозованих значень інтенсивності навантаження можливо здійснити адаптацію шаблонів масштабування, що дозволить уникнути порушення вимог щодо доступності системи у випадку динамічних змін навантаження.

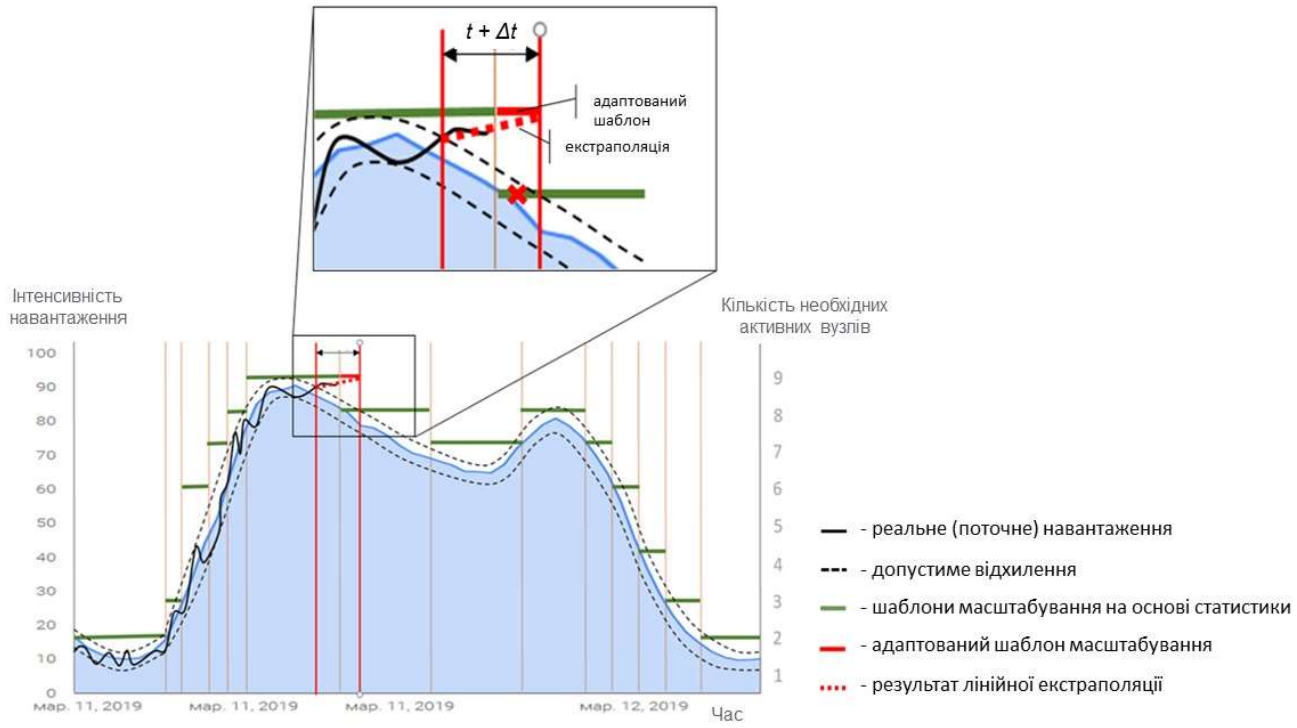


Рисунок 3.4 Основна ідея підходу динамічного керування ресурсами

Підсумуємо кроки запропонованого комплексного методу, їх реалізацію та вплив на окремі визначені показники ефективності та критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max$ при $p_{avail} \geq p_{ava\ \ SLA}$ у таблиці (Таблиця 3.1).

Таблиця 3.1 Реалізація кроків запропонованого комплексного методу та їх вплив на визначені показники ефективності

Крок	**Реалізація кроку (підсумок)**	**Вплив на визначені показники ефективності**
Атестація та сортування обчислювальних вузлів	– Визначити показник енергоспоживання кожного вузла $S_j = \int_0^{100} P_j(CPU_j)dCPU_j$. – Виконати сортування вузлів за показником енергоспоживання S_j. – Виконати сортування вузлів за номінальним значенням показника їх продуктивності C_j. ➔ <u>Результат</u>: Ранжований список вузлів від найгіршого до найкращого за показниками S_j та C_j.	Інтенсивніше використання вузлів з кращими показниками S_j та C_j веде до збільшення показника сумарної продуктивності системи C_Σ та зниження сумарного енергоспоживання системи $W = \sum_{j=1}^{N} W_j$ при обробці навантаження ω. → $\boldsymbol{K_{opt} = E_\Sigma * C_\Sigma \to max}$
Метод визначення шаблонів горизонтального масштабування	– При заданій максимально допустимій імовірності втрати запиту $p_{loss_{SLA}}$ визначити порогові значення інтенсивності вхідного навантаження $\lambda(t)_{\text{п}} = \mu' * \sqrt[n+Q]{\frac{p_{loss_{SLA}} * n^Q * n!}{p_0}}$, $n \in \mathbb{N}^*$ – Для добової статистики навантаження для кожного з інтервалів Δt, що відповідають переходу від значення $\lambda(t)_{\text{п},i}$ до $\lambda(t)_{\text{п},i+1}$, визначити $\lambda' = max(\lambda_i), \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$. – Для кожного з інтервалів Δt визначити *мінімально необхідну* кількість обчислювальних вузлів n, при якій нерівність $p_{loss} = \frac{(\frac{\lambda'}{\mu'})^{n+Q}}{n^Q * n!} p_0 < p_{loss_{SLA}}$ буде справджуватись при $\lambda' = max(\lambda_i), \lambda_i \in [\lambda(t)_{\text{п},i}, \lambda(t)_{\text{п},i+1}]$. ➔ <u>Результат:</u> Мінімально необхідна кількість активних обчислювальних вузлів n на інтервалі часу Δt для дотримання вимог щодо доступності системи $p_{avail} \geq p_{avail_{SLA}}$ при зниженні сумарної потужності споживання $P_\Sigma = \sum_{j=1}^{n} P_j\left(CPU_{avg_j}\right) \to min$	Визначена мінімально необхідна кількість активних вузлів n для інтервалу часу Δt дозволить перевести $(N - n)$ вузлів у режим сну, тим самим зменшуючи сумарну потужність споживання P_Σ та енергоспоживання системи $W = \sum_{j=1}^{n} W_j$. Врахування обмеження $p_{avail} \geq p_{avail_{SLA}}$ при визначенні n забезпечує виконання вимог щодо доступності системи. → $\boldsymbol{E_\Sigma \to max}$ → $\boldsymbol{p_{avail} \geq p_{avail_{SLA}}}$

Таблиця 3.1 (продовження)

Крок	**Реалізація кроку (підсумок)**	**Вплив на визначені показники ефективності**
Застосування визначених шаблонів для горизонтального масштабування та енергоефективного планування задач	– Для кожного інтервалу часу Δt виконати переведення $(N-n)$ вузлів системи у режим сну. При цьому n активних вузлів обираються з ранжованого списку згідно з показниками S_j та C_j (див. Крок 1). – Для розподілу обчислювальних робіт між вузлами використати алгоритм розподілу навантаження Backfill. ➔ <u>Результат</u>: Мінімізація сумарної потужності споживання системи $P_\Sigma = \sum_{j=1}^{n} P_j\left(CPU_{avg_j}\right) \to min$ шляхом переведення $(N-n)$ найменш ефективних за показниками S_j та C_j вузлів у режим сну. Максимізація продуктивності системи за рахунок ущільнення навантаження ($CPU_j \to max, T \to min$ при $\omega = const$).	Мінімізація сумарної потужності споживання системи P_Σ веде до зменшення сумарного енергоспоживання системи $W = \sum_{j=1}^{n} W_j$ та як наслідок збільшення показника енергоефективності E_Σ при сталому обсязі навантаження ω. Використання алгоритму Backfill забезпечує обробку обсягу навантаження ω за менший час T за рахунок щільного розміщення задач на вузлах, тобто $CPU_j \to max$, що спричиняє зростання продуктивності C_Σ. →$\boldsymbol{K_{opt} = E_\Sigma * C_\Sigma \to max}$
Виявлення відхилення поточного навантаження від статистичного та динамічна адаптація шаблонів масштабування	– У момент t при виявленні відхилення інтенсивності поточного навантаження $\lambda_{obs}(t)$ від статистичного $\lambda_{pred}(t)$ на величину більшу σ розрахувати величину вікна адаптації: – $W(t) = I_{base} - K * \frac{\sum_{j=t-h}^{t-1} max(\sigma; \lvert\lambda_{obs}(j) - \lambda_{pred}(j)\rvert)}{h-1} I_{base}$ – Шляхом екстраполяції визначити прогнозовану інтенсивність навантаження з урахуванням відхилення у момент часу $(t+\Delta t)$, що відповідає наступному моменту зміни шаблону масштабування: – $\lambda_{pred}(t+\Delta t) = \lambda_{basepred}(t+\Delta t) + \sum_{j=t-h}^{t-1} \frac{\lambda_{obs}(j) - \lambda_{pred}(j)}{h-1}$ – Здійснити перерахунок шаблону масштабування при $\lambda' = \lambda_{pred}(t+\Delta t)$. – Повторити процедуру у момент часу $(t + W(t))$. ➔ <u>Результат</u>: Адаптація шаблону масштабування відповідає динамічно зміненій інтенсивності вхідного навантаження $\lambda_{pred}(t+\Delta t)$. Вимоги до доступності системи $p_{avail} \geq p_{avail_{SLA}}$ не порушені.	Адаптація шаблонів масштабування дозволяє підтримувати їх актуальними при динамічно змінюваній інтенсивності вхідного навантаження $\lambda_{obs}(t)$ та уникнути порушення вимог до доступності системи. →$\boldsymbol{p_{avail} \geq p_{avail_{SLA}}}$

ВИСНОВКИ

1. Запропоновано комплексний метод обслуговування навантаження ІКМ, що дозволяє підвищити енергоефективність E_{Σ} та продуктивність C_{Σ} обслуговування навантаження з одночасним урахуванням вимог щодо доступності системи. Метод враховує вимоги щодо обслуговування навантаження ІКМ та базується на рекомендаціях ITU-T L.1300 Міжнародної спілки електрозв'язку. На відміну від існуючих підходів, запропонований комплексний метод враховує індивідуальні моделі енергоспоживання обчислювальних вузлів, поєднує переваги підходів горизонтального масштабування та енергоефективного розподілу задач, враховує можливість непередбачуваних динамічних змін інтенсивності вхідного навантаження, що є науковою новизною.

2. У рамках запропонованого комплексного методу вдосконалено існуючі підходи щодо горизонтального масштабування обчислювальної системи шляхом оцінки індивідуальних моделей енергоспоживання обчислювальних вузлів, що дозволило забезпечити інтенсивніше використання найбільш енергоефективних вузлів в процесі обслуговування навантаження.

3. Вперше запропоновано метод розрахунку шаблонів горизонтального масштабування обчислювальної системи, що базується на побудованій моделі системи як СМО. Метод дозволяє визначати оптимальну кількість активних обчислювальних вузлів у системі на кожному інтервалі часу, що визначається швидкістю зміни інтенсивності вхідного навантаження. За рахунок цього у кожний момент часу кількість активних вузлів у системі є достатньою для обробки вхідного навантаження при дотриманні вимог щодо імовірності втрати запиту, але мінімальною, що забезпечує зниження сумарного енергоспоживання системи.

4. Існуючі підходи щодо горизонтального масштабування удосконалено шляхом застосування механізму прогнозування динамічних відхилень вхідного навантаження, що дозволяє прогнозувати та вчасно реагувати на неочікувані зміни інтенсивності вхідного навантаження з метою дотримання вимог щодо доступності системи та продуктивності обслуговування навантаження ІКМ.

РОЗДІЛ 4
АНАЛІЗ ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО КОМПЛЕКСНОГО МЕТОДУ ЕНЕРГОЕФЕКТИВНОГО ОБСЛУГОВУВАННЯ НАВАНТАЖЕННЯ

4.1 Опис розроблених програмних засобів керування обчислювальними ресурсами

На основі запропонованого комплексного методу було розроблено програмне забезпечення керування обчислювальними ресурсами. Компонентна модель розробленого ПЗ зображена на рисунку (Рисунок 4.1).

Розроблене ПЗ складається з таких основних компонентів:

– головний вузол, що включає:

- компонент атестації;
- менеджер ресурсів;
- компонент горизонтального масштабування;
- компонент планування навантаження;
- компонент моніторингу і прогнозування навантаження;

– вузол обробки (робочий вузол).

Компонент атестації

Компонент атестації (або блок атестації) обробляє атестаційні дані робочих вузлів (отримані експериментально або програмним шляхом на основі відомих бенчмарків) і будує функції енергоспоживання вузлів в аналітичній формі. Основна мета цього вузла – отримати рейтинг вузлів, відсортованих за критеріями енергоефективності та продуктивності.

Менеджер ресурсів і компонент горизонтального масштабування

Менеджер ресурсів і компонент горизонтального масштабування працюють разом, щоб вирішити спільне завдання горизонтального масштабування. Менеджер ресурсів заздалегідь розраховує шаблони масштабування для системи на основі статистичних даних про вхідне навантаження та перераховує їх, якщо необхідно. Він вирішує, скільки активних вузлів має бути залучено до обробки, щоб відповідати вимогам доступності системи згідно з SLA та забезпечити мінімальне

енергоспоживання. Компонент горизонтального масштабування використовує побудовані шаблони масштабування та переводить певну кількість вузлів у режим сну під час роботи системи.

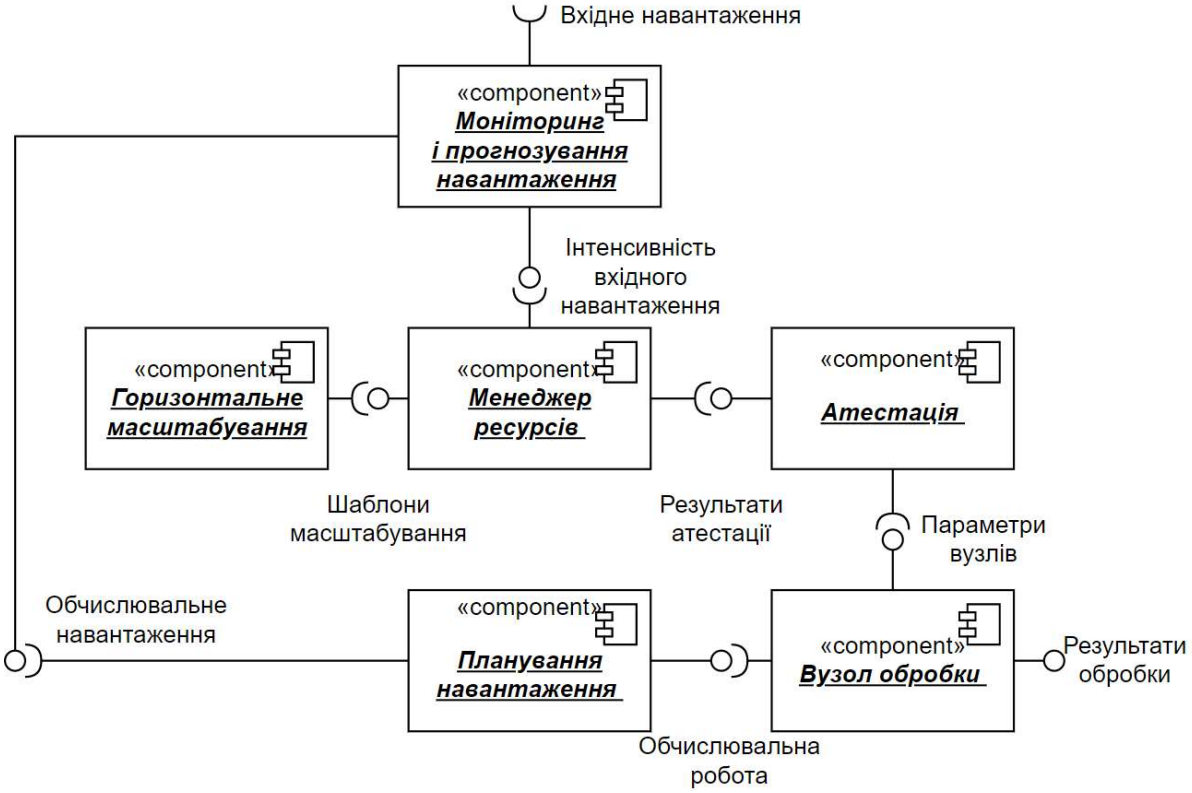


Рисунок 4.1 Компонентна архітектура ПЗ керування обчислювальними ресурсами на основі запропонованого комплексного методу

Компонент планування навантаження

Цей компонент розподіляє навантаження між вузлами обробки за допомогою алгоритму планування навантаження Backfill, що забезпечує максимальний рівень використання активних ресурсів. За потреби інші алгоритми планування навантаження також можуть бути використані, якщо вони мають відповідні інтерфейси для взіємодії із іншими компонентами запропонованого ПЗ.

Компонент моніторингу і прогнозування навантаження

Компонент моніторингу і прогнозування навантаження реалізує четвертий крок запропонованого комплексного методу. Він відстежує поточну інтенсивність надходження робочого навантаження та порівнює її зі статистичною на відповідному інтервалі часу. Він оцінює поточне відхилення інтенсивності

надходження робочого навантаження від статистичного та прогнозує можливі несподівані стрибки або зменшення робочого навантаження за допомогою методу лінійної екстраполяції. Він інформує менеджера ресурсів, якщо є потреба оновити шаблони масштабування. Компонент моніторингу і прогнозування навантаження має вхідний інтерфейс «Вхідне навантаження», що означає, що цей компонент отримує вхідні дані про поточне навантаження ззовні.

Вузол обробки (робочий вузол)

Вузли обробки призначені для безпосередньої обробки обчислювальних робіт. При необхідності їх можна переводити в режим сну і повертати в активний режим в рамках горизонтального масштабування системи. Вузол обробки надає інтерфейс «Результати обробки», що означає, що результати обробки обчислювальних робіт повертаються користувачеві як відповідь.

У тестовій системі, створеній для експерименту, було створено додатковий вузол ***«Клієнт»***. У реальній системі вхідне робоче навантаження надходить у систему від великої кількості територіально розподілених вузлів. Однак у створеному тестовому програмному забезпеченні клієнтський вузол виступає як абстракція всіх джерел вхідного навантаження. З нього обчислювальні завдання надходять в систему до Компонента моніторингу і прогнозування навантаження для їх подальшого аналізу, розподілу та обробки.

4.2 Застосування розроблених програмних засобів в інформаційно-комунікаційній мережі

В розглянутій у пункті 1.1.1 (Рисунок 1.1) архітектурі ІКМ 5G розроблене ПЗ може бути використане у розподілених ЦОД периферійної хмари (Edge Computing) та центральної хмари (Core/Central Cloud) (Рисунок 4.2).

При використанні ***у структурі периферійної хмари*** джерелами навантаження для розробленого програмного забезпечення є безпосередньо кінцеві пристрої. Це, зокрема:

– системи відеоспостереження (задачі аналізу відео та детектування позаштатних ситуацій у режимі реального часу);

– користувацькі додатки (задачі локального кешування контенту, аналізу даних від пристроїв Інтернету речей (IoT, від англ. Internet of Things)), тощо.

Вузлами обробки при цьому виступають як сервери у невеликих децентралізованих ЦОД, територіально близьких до користувача, так і самі периферійні пристрої (згідно з концепцією розподілених обчислень з використанням кінцевих пристроїв користувачів) [127]. При цьому, у випадку використання користувацьких пристроїв у якості обчислювальних вузлів периферійної хмари, актуальність задачі забезпечення енергоефективності таких обчислень суттєво зростає, адже такі пристрої часто живляться від акумуляторних батарей.

Таким чином, для інтеграції розробленого ПЗ з інфраструктурою периферійної хмари у ній необхідно виділити 2 додаткові логічні елементи, що відповідають функціональним вимогам головного вузла та вузлів обробки.

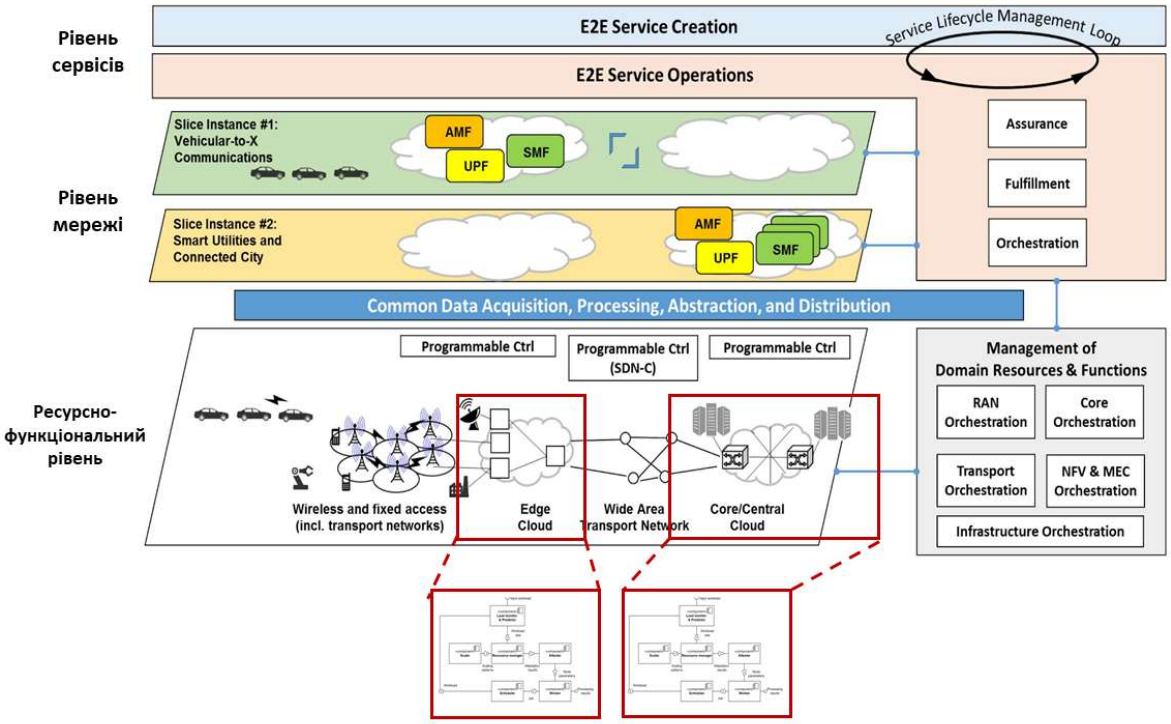


Рисунок 4.2 Місце розробленого ПЗ в архітектурі інформаційно-комунікаційної мережі 5G

При використанні розробленого ПЗ в архітектурі ***центральної хмари*** робочим навантаженням зокрема виступають:

– задачі віртуалізованих мережевих функцій у рамках концепції NFV (задачі маршрутизації, комутації, перетворення мережевих адрес);

– задачі оркестрації та реконфігурації мережевих ресурсів у рамках SDN;

– задачі виокремлення логічних рівнів мережі, аналізу вимог сервісів на кожному з логічних рівнів, моніторингу виконання вимог щодо якості сервісу на кожному рівні згідно з концепцією Network Slicing;

– задачі обробки Великих даних (про поточну конфігурацію мережі, стан мережевого обладнання, статистику мережевого трафіку, історію помилок та збоїв у мережі, кількість абонентів, що обслуговуються базовими станціями, тощо) у рамках bDDN.

Вузлами обробки виступають сервери розподілених ЦОД, що об’єднані у окремі обчислювальні кластери. При цьому, розроблене ПЗ може бути розміщене на центральному вузлі-брокері у кожному кластері.

Таким чином, розроблене програмне забезпечення керування обчислювальними ресурсами рекомендовано до використання у розподілених ЦОД периферійної хмари та центральної хмари в інформаційно-комунікаційній мережі 5G.

4.3 Оцінка ефективності функціонування розробленого програмного забезпечення керування обчислювальними ресурсами

З метою оцінки ефективності запропонованого комплексного методу та розробленого на його основі програмного забезпечення керування обчислювальними ресурсами було проведено лабораторний експеримент, а також імітаційне моделювання із застосуванням мови програмування Python. Розглянемо докладно процес та результати експерименту й імітаційного моделювання, окремі аспекти та результати яких було опубліковано у працях [48-51].

4.3.1 Експериментальне дослідження ефективності запропонованого комплексного методу

Для оцінки ефективності запропонованого комплексного методу на реальному серверному обладнанні, було проведено лабораторний експеримент на базі серверного кластера лабораторії Технічного університету м. Дрездена (Німеччина).

4.3.1.1 Постановка експерименту

Для проведення експерименту було використано 4 сервери з такими іменами та IP-адресами відповідно:

– Server_1: 141.76.65.51;

– Server_2: 141.76.65.52;

– Server_3: 172.22.1.167;

– Server_4: 141.76.65.28;

що виконували роль обчислювальних вузлів, та клієнтську ЕОМ для виконання функції головного вузла та вузла моніторингу. Наявне обладнання було організовано у комп'ютерний кластер із топологією зірки, параметри обчислювальних вузлів наведено на рисунку (Рисунок 4.3). Продуктивність вузлів на рисунку виражено у відносних одиницях Pt (від англ. point), значення продуктивності взято із джерела [124] для відповідних моделей процесорних вузлів. Як вже згадувалось, використання відносних одиниць є допустимим в експерименті, оскільки абсолютні значення продуктивності не використовуються у запропонованому комплексному методі.

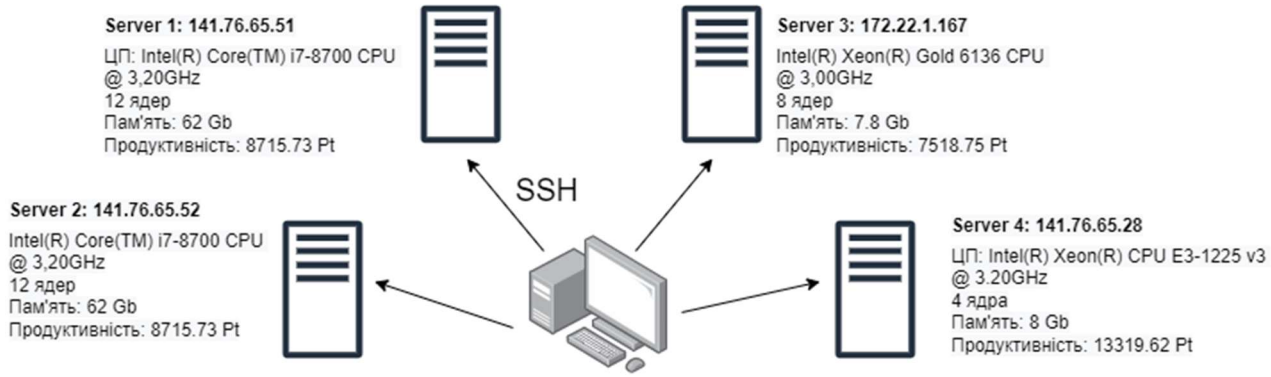


Рисунок 4.3 Досліджуваний в експерименті комп'ютерний кластер

На обчислювальних вузлах було встановлено серверну операційну систему Ubuntu Server (дистрибутив GNU/Linux, тобто операційна система, що використовує ядро Linux та є готовою для установки на користувацьке обладнання). Віддалене підключення від головного вузла до обчислювальних вузлів здійснювалось із використанням протоколу *SSH* (від англ. Secure Shell) – мережевий протокол рівня додатків, що дозволяє проводити віддалене керування комп'ютером і тунелювання TCP-з'єднань. На кожному з обчислювальних вузлів здійснювався моніторинг поточної завантаженості процесора за допомогою утиліти *vmstat* – ПЗ моніторингу активності ЦП, стану елементів пам’яті, системи введення/виведення.

Необхідним кроком для формулювання вхідних даних для експерименту згідно із запропонованим комплексним методом є визначення індивідуальних залежностей енергоспоживання обчислювальних вузлів системи. Для цього з метою порівняння було використано:

1. Експериментальне визначення енергоспоживання безпосередньо у колі живлення серверів з використанням інтерполяції отриманих значень поліномом ступеня n.
2. Програмне визначення**:**
   - з подальшою апроксимацією лінійною функцією;
   - з інтерполяцією отриманих значень поліномом ступеня n.

В процесі експериментального визначення енергоспоживання кожен з обчислювальних вузлів розглядається окремо (без урахування топології кластера). Для зняття характеристик $P_j = f_j(CPU_j)$ кожного вузла, було використано цифровий мультиметр Yokogawa WT210, що здатний вимірювати струм, напругу та миттєву потужність безпосередньо між джерелом живлення та навантаженням. Для зняття характеристик $P_j = f_j(CPU_j)$ було використано навантажувальний тест (англ. stress-test) [128] (можливим є також використання інших навантажувальних тестів, наприклад [129]). Зняття характеристик проводилось шляхом послідовного

підключення мультиметра у коло живлення кожного сервера. Вимірювання потужності споживання проводилось для $CPU_j = (0, 10, \ldots, 100\%)$ завантаженості із кроком 10%. Отримані у результаті табличні функції енергоспоживання наведені у додатках (Додаток 1).

Робоче навантаження на кластер було представлено за допомогою 3 типів обчислювальних робіт, із різними вимогами щодо кількості обчислювальних ядер:

1. ***Алгоритм KMeans*** [130] – широко відомий алгоритм кластеризації за методом k-середніх. Метою алгоритму є розбиття набору точок на групи, що називають кластерами, при мінімізації сумарного квадратичного відхилення точок кластерів від центрів цих кластерів. Алгоритм широко використовується зокрема при вирішенні задач машинного навчання.

2. ***Алгоритм Sudokount*** [131] – запропонований Петером Норвігом (Peter Norvig) у [132] алгоритм для вирішення будь-якого варіанту головоломки Sudoku – логічної японської головоломки з числами. Для розв’язання головоломки необхідно заповнити пусті клітинки поля Sudoku послідовними натуральними числами так, щоб числа не повторювались у рядку, стовпці та клітинці. В основі алгоритму лежать принципи так званих “поширення обмежень” та “пошуку”.

3. ***Алгоритм Game of Life*** [131] – це алгоритм моделювання так званої “гри життя” (англ. Game of Life). “Гра життя” – клітинний автомат, запропонований англійським математиком Джоном Конвеєм у 1970 році. Місце дії гри являє собою площину, поділену на клітинки. Кожна клітинка може перебувати в одному з двох станів: бути “живою” або “мертвою” (темною або світлою). Кожна клітинка має вісім сусідів. Розподіл “живих” клітинок на початку гри називається першим поколінням. Кожне наступне покоління утворюється на основі попереднього за певними правилами, що детально описано, наприклад, у джерелі [133].

Ці алгоритми було обрано для використання в експерименті тому, що вони представляють різні за часом виконання обчислювальні задачі та можуть бути легко розпаралелені. Це дозволило отримати різні за складністю та вимогами до обчислювальних ресурсів задачі.

Кожен з цих алгоритмів реалізовано на мові програмування C з використанням бібліотеки *pthreads* [134], що імплементує стандарт POSIX реалізації нитей виконання, який визначає API для створення та керування багатопоточними програмами. Тобто, реалізація кожного з алгоритмів передбачає можливість їх виконання на довільній кількості ядер процесора. Реалізація кожного з алгоритмів на певній кількості ядер розглядається як окрема обчислювальна робота, що має відповідні вимоги до кількості ядер $cores_{min_i}$ та відповідний розрахунковий час виконання Δt_{max_i}(розрахунковий час виконання визначено шляхом послідовного вимірювання часу виконання на сервері 1 (Server_1: 141.76.65.51), що має середні характеристики щодо продуктивності). Таким чином, отримано набір з 18 типів обчислювальних робіт (Таблиця 4.1), де у строках наведено час виконання відповідної задачі на 1,...,24 ядрах у секундах. Кожне вимірювання було проведено 5 разів, у таблиці наведені середні отримані значення для часу виконання.

Таблиця 4.1 Обчислювальні роботи, використані у якості навантаження, та їх середній час виконання на різній кількості ядер

Тип обчисл. роботи / Кількість ядер	**1**	**2**	**4**	**8**	**12**	**24**
KMeans	59.120000 с	29.560000 с	16.891428 с	11.824000 с	7.390000 с	5.912000 с
Sudokount	279.207066 с	144.053876 с	79.856665 с	43.597932 с	39.886723 с	27.920706 с
Game of Life	3.141487 с	2.121443 с	1.994767 с	1.980366 с	1.970366 с	1.960366 с

Оскільки обчислювальні роботи використані в ході експерименту мають різну складність та час виконання, використання параметру ω (ω – кількість успішно оброблених обчислювальних робіт) у процесі оцінки ефективності запропонованого комплексного методу є некоректним. Для більш точного визначення одиниці навантаження, яка може оцінюватись у рамках оцінки ефективності запропонованого комплексного методу, було проведено нормалізацію кількості обчислювальних робіт за часом їх виконання на 1 ядрі (Таблиця 4.2). Нормалізацію проведено за формулою (4.1).

$$\omega' = \sum_{i=1}^{3} \omega_i * h, \tag{4.1}$$

де ω_i – кількість успішно оброблених обчислювальних робіт i-го типу, $i \in [KMeans, Sudokount, Game\ of\ Life]$; $h = \Delta t_{1core}/\Delta t_{1core-min}$ – коефіцієнт нормалізації, що визначається відношенням середнього часу виконання обчислювальної роботи та найменшого середнього часу виконання обчислювальної роботи серед наявних типів робіт та відображає приведений “розмір” задачі.

Таблиця 4.2 Результат нормалізації обчислювальних робіт за часом виконання

Тип обчислювальної роботи	**Час виконання на 1 ядрі** Δt_{1core}	**Приведений “розмір” роботи** $h = \Delta t_{1core}/\Delta t_{1core-min}$
KMeans	59,120000 с	18,819
Sudokount	279,207066 с	88,877
Game of Life	3,141487 с	1

Таким чином, надалі для оцінки продуктивності та енергоефективності системи замість параметра ω використовується приведений параметр ω' (ω' – приведена кількість успішно оброблених обчислювальних робіт), що точніше відображає обсяг обробленого навантаження.

У якості статистики робочого обчислювального навантаження була використана крива зображена на рисунку (Рисунок 4.4), побудована на основі статистики добового навантаження на ЦОД з джерела [135]. Дискретні значення інтенсивності вхідного навантаження, що відповідають точкам кривої, отримано за допомогою формули (2.8).

Вхідні дані, використані в експерименті наведено у таблиці (Таблиця 4.3).

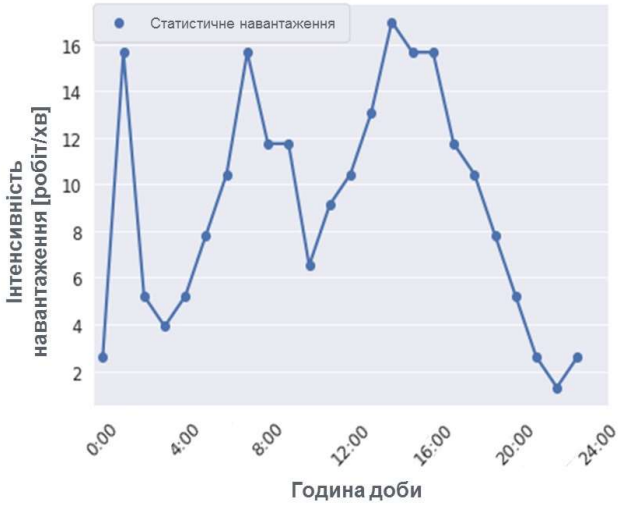


Рисунок 4.4 Статистична крива навантаження, використана в експерименті [135]

Таблиця 4.3 Вхідні дані, використані в експерименті

Параметр	**Опис**
$n=4$	Кількість обчислювальних вузлів
C_j = {8715.73, 8715.73, 7518.75, 13319.62} Pt	Номінальна продуктивність вузлів
$Q=5$	Довжина черги [обчислювальних робіт або заявок СМО]
$CPU_{max_j}=75\%$	Середня завантаженість ЦП вузлів при використанні Backfill [125]
$\mu'_{avg}= 0,03$	Середня швидкість обробки заявок [обчислювальних робіт на секунду] (на 1 ядрі)
$p_{loss_SLA}=5*10^{-2}$	Максимально допустима ймовірність втрати заявки
$n'=7*4=28$	Кількість обслуговуючих пристроїв СМО (середня кількість ядер вузла – 7)

Експеримент було проведено за таким планом:

1. Оцінка запропонованого комплексного методу:

– підготовча фаза (атестація та сортування вузлів, визначення шаблонів масштабування);

– основна фаза (горизонтальне масштабування та планування навантаження, прогнозування відхилень навантаження та адаптація шаблонів).

2. Оцінка запропонованого комплексного методу без вимкнення вузлів:

– підготовча фаза (атестація та сортування вузлів, визначення шаблонів масштабування);

– основна фаза (горизонтальне масштабування та планування навантаження, прогнозування відхилень навантаження та адаптація шаблонів).

3. Оцінка енергоефективного алгоритму планування Backfill (порівняння із запропонованим комплексним методом);

4. Оцінка алгоритму планування Round Robin (порівняння із запропонованим комплексним методом).

Для оцінки ефективності підходів було використано показники:

1. Показник енергоефективності: $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j} [\frac{\text{робіт}}{\text{Вт*год}}]$, де ω' – приведена кількість успішно оброблених обчислювальних робіт; W_j – сумарний обсяг електроенергії спожитої j-им вузлом; n – кількість активних вузлів у системі.

2. Показник продуктивності: $C_{\Sigma} = \frac{\omega'}{T} [\frac{\text{робіт}}{\text{год}}]$, де ω'– приведена кількість успішно оброблених обчислювальних робіт; T – тривалість обробки.

3. Коефіцієнт готовності системи: $p_{avail} = 1 - p_{loss}$, де p_{loss} – імовірність втрати запиту.

та критерій ефективності: $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$.

4.3.1.2 Оцінка ефективності запропонованого комплексного методу

Згідно з запропонованим комплексним методом, першим кроком є визначення моделей енергоспоживання обчислювальних вузлів у досліджуваній системі (атестація вузлів). В рамках лабораторного експерименту було використано та порівняно 3 підходи щодо визначення моделей енергоспоживання:

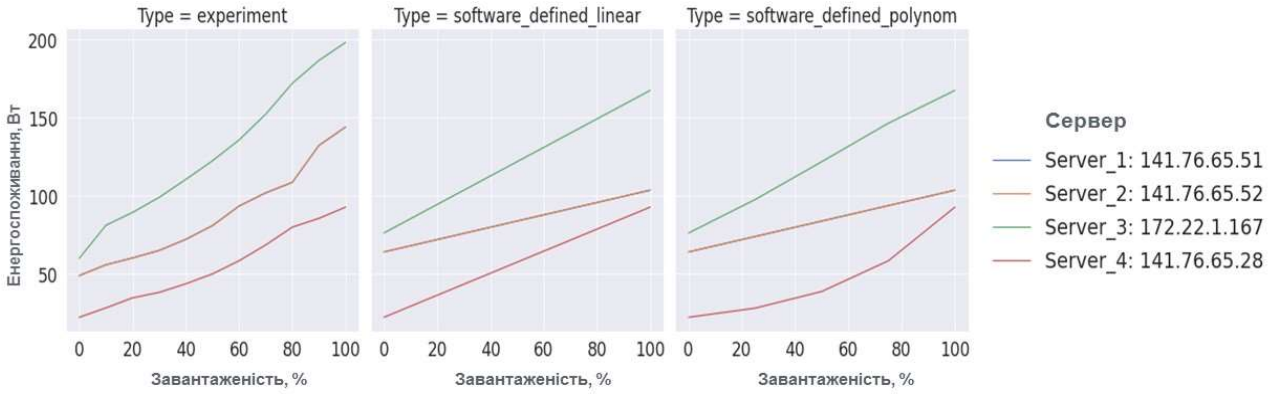
1. Експериментальне визначення енергоспоживання безпосередньо у колі живлення серверів (цей підхід описано у пункті 2.3.1) з використанням інтерполяції отриманих значень поліномом ступеня n;

2. Програмне визначення, описане у пункті 2.3.2:

– з подальшою апроксимацією лінійною функцією;

– з інтерполяцією отриманих значень поліномом ступеня n.

У результаті було отримано моделі енергоспоживання для кожного сервера, зображені на рисунку (Рисунок 4.5). Оскільки два з чотирьох серверів мають однакові характеристики АЗ, їх моделі енергоспоживання практично співпадають.



а) б) в)

Рисунок 4.5 Моделі енергоспоживання серверів використаних в експерименті

а) експериментально визначені залежності;	б) залежності визначені програмним шляхом із використанням лінійної апроксимації;	в) залежності визначені програмним шляхом із використанням інтерполяції поліномом

У випадку застосування інтерполяції залежності розраховувались за формулою (3.2). Отримані функції у вигляді поліномів зображені на рисунку (Рисунок 4.6).

```
-----------------P(CPU) functions-------------------
For worker natgvo@141.76.65.51 P(CPU)=-9.08E-8x^4 + 5.39E-5x^3 + -4.59E-5x^2 + 0.51x + 49.55
Integral: 8653.797898008417
For worker natgvo@141.76.65.52 P(CPU)=-9.08E-8x^4 + 5.39E-5x^3 + -4.59E-5x^2 + 0.51x + 49.55
Integral: 8653.797898008417
For worker prokopets@172.22.1.167 P(CPU)=-4.59E-6x^4 + 9.41E-4x^3 + -5.76E-2x^2 + 2.32x + 60.72
Integral: 12787.215099715087
For worker nagv@141.76.65.28 P(CPU)=-2.29E-6x^4 + 4.34E-4x^3 + -2.22E-2x^2 + 0.88x + 22.2
Integral: 5465.060217560211
```

Рисунок 4.6 Отримані функції у вигляді поліномів

Для порівняння серверів у рамках запропонованого комплексного методу було підраховано визначені інтеграли отриманих функції енергоспоживання на проміжку від 0 до 100% (що відповідає завантаженості процесора) за формулою (3.3).

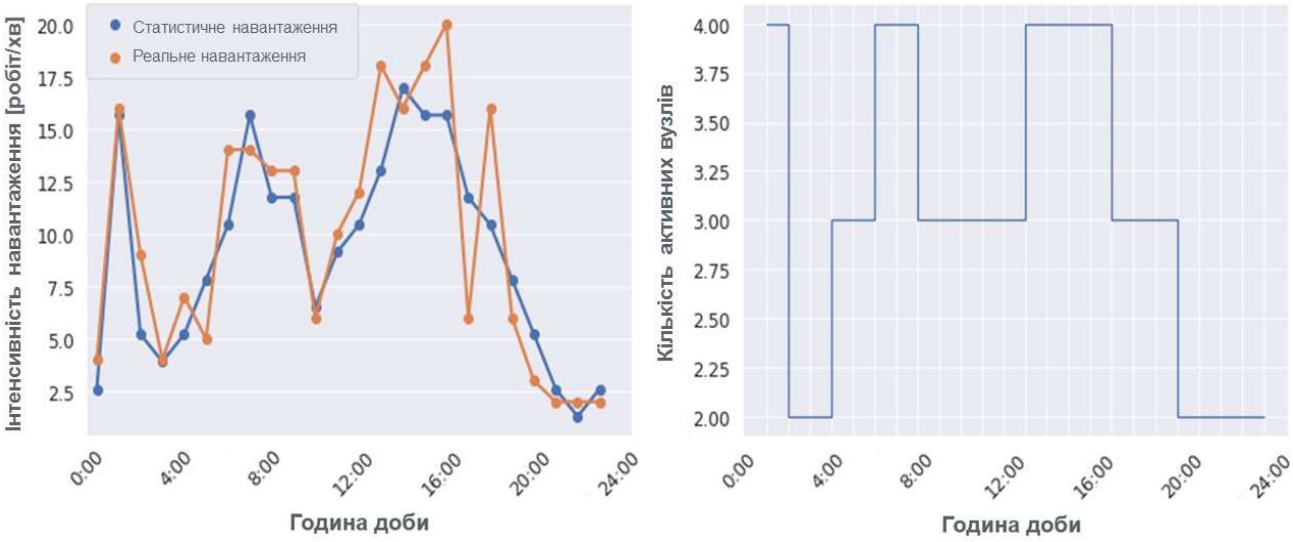
Обчислювальні вузли було посортовано за значеннями, отриманими за допомогою інтегралів (3.3), а також за номінальними значеннями продуктивності вузлів. У результаті кожному вузлу було проставлено оцінки згідно його положення у сортованих за продуктивністю та енергоспоживанням масивах. Вузол із найвищою сумарною оцінкою вважався відповідно найкращим за критеріями продуктивності та енергоефективності. В результаті отримано послідовність обчислювальних вузлів від найкращого до найгіршого (Таблиця 4.4).

Таблиця 4.4 Результат атестації та сортування обчислювальних вузлів

Сервер	**Сумарна оцінка**	**Оцінка за енергоефективність**	**Оцінка за продуктивність**
Server_4: 141.76.65.28	6	3	3
Server_1: 141.76.65.51	4	2	2
Server_2: 141.76.65.52	4	2	2
Server_3: 172.22.1.167	2	1	1

Таким чином, в результаті процесу атестації визначено, що вузол *“Server_4: 41.76.65.28”* повинен використовуватись якомога інтенсивніше під час процесу обробки, у той час як вузол *“Server_3: 172.22.1.167”* навпаки слід за можливості якомога більше часу тримати у стані сну, оскільки він споживає більше енергії при меншій продуктивності.

Наступним етапом є визначення шаблонів горизонтального масштабування досліджуваної системи. Розрахунок шаблонів для кожного інтервалу часу та розрахунок оптимальних інтервалів відповідно було здійснено із використанням методу запропонованого у пункті 3.1.2. Визначені шаблони для вхідного обчислювального навантаження зображено на рисунку (Рисунок 4.7 (б)) – по осі ординат відкладено оптимальну кількість вузлів, а по осі абсцис відображена довжина інтервалів часу між змінами шаблонів.

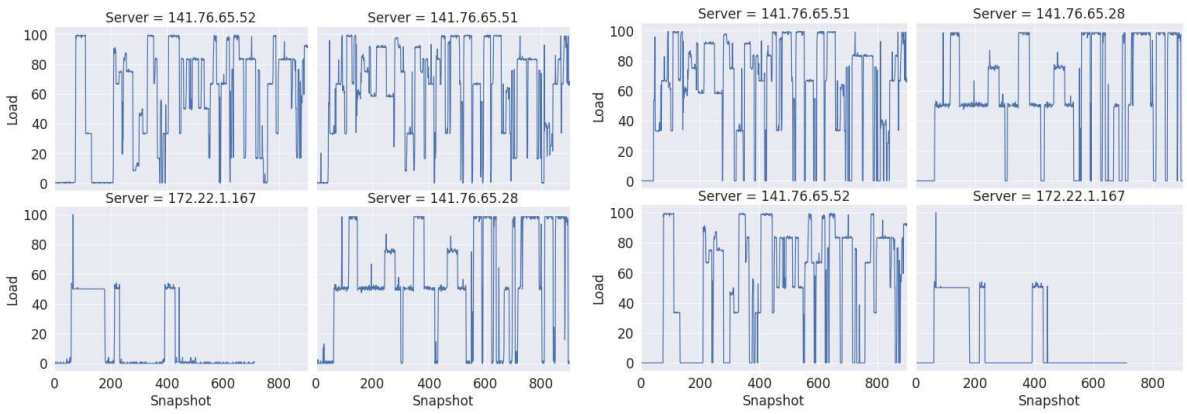


а) б)

Рисунок 4.7 Статистика інтенсивності вхідного навантаження та шаблони горизонтального масштабування визначені в експерименті:

а) вхідне обчислювальне навантаження: синя крива – статистичне навантаження, помаранчева крива – реальне навантаження з відхиленнями; б) визначені шаблони масштабування.

У процесі роботи системи частина вузлів (найгірші за показниками енергоспоживання та продуктивності) переводилась у режим сну згідно визначених шаблонів з метою зменшення сумарного енергоспоживання системи. Як вже згадувалось, у процесі обробки обчислювального навантаження здійснювався моніторинг поточної завантаженості обчислювальних вузлів за допомогою утиліти *vmstat*. У результаті було отримано криві завантаженості для кожного вузла (Рисунок 4.8). Додатково було порівняно варіанти, коли вузли переводились в режим сну та коли вони залишались у режимі простою у випадку, якщо згідно з визначеними шаблонами на них не подавалось навантаження для обробки (Рисунок 4.8 (а) та (б) відповідно).



а) б)

Рисунок 4.8 Криві поточної завантаженості обчислювальних вузлів в ході експерименту:

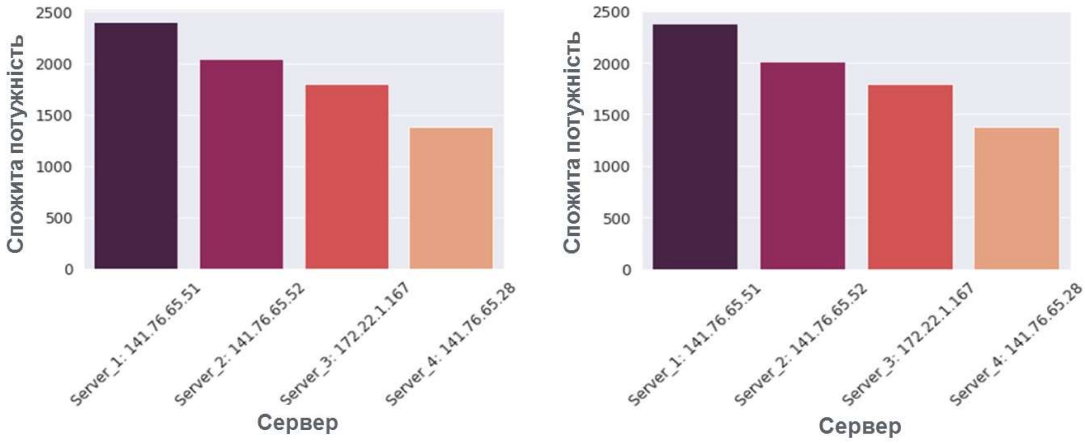
а) без переведення серверів в режим сну; б) з переведенням в режим сну згідно шаблонів.

Як видно з рисунка, сервер *“Server_3: 172.22.1.167”* був найменш завантаженим в процесі обробки, що відповідає цілям запропонованого комплексного методу, адже цей вузол є найменш ефективним як за параметром продуктивності, так і за енергоспоживанням. На графіках також прослідковуються періоди переведення вузлів у режим сну, коли їх завантаженість на певний час падає до 0. Варто відзначити також, що завантаженість вузлів *“Server_1: 141.76.65.51”, “Server_2: 141.76.65.52”* та *“Server_4: 141.76.65.28”* загалом була рівномірною та підтримувалась на високому рівні. Це пояснюється тим, що завдяки використанню алгоритма Backfill задачі розміщувались на вузли так, щоб досягти максимальної завантаженості кожного вузла у кожен момент часу, адже режим роботи при високих рівнях завантаженості є більш ефективним.

На основі даних про поточну завантаженість вузлів в ході експерименту, а також моделей енергоспоживання вузлів, було визначено сумарне енергоспоживання кожного вузла (Рисунок 4.9) та системи в цілому за час експерименту. Для цього було використано формулу (4.2).

$$W = \sum_{t=1}^{24} P_j(\overline{CPU_{jt}}) \tag{4.2}$$

де $\overline{CPU_{jt}}$ – середня завантаженість процесора j-го вузла протягом t-ої години експерименту; $P_j(\overline{CPU_{jt}})$ – функція енергоспоживання j-го обчислювального вузла.



а) б)

Рисунок 4.9 Порівняння сумарного енергоспоживання вузлів в ході експерименту:

а) без переведення серверів в режим сну; б) з переведенням в режим сну згідно шаблонів.

У таблиці (Таблиця 4.5) наведено підсумок результатів експериментальної перевірки запропонованого комплексного методу щодо оцінюваних показників.

4.3.1.1 Оцінка ефективності енергоефективного алгоритму планування Backfill (порівняння із запропонованим комплексним методом)

Наступним кроком було проведено оцінку енергоефективного алгоритму планування навантаження Backfill. У цьому випадку не було необхідності у проведенні етапу атестації та сортування обчислювальних вузлів, а також у побудові шаблонів масштабування. Тому, експериментальна перевірка цього підходу полягала у генерації, розподілі та обробці обчислювальних робіт протягом усього часу експерименту. У процесі обробки обчислювального навантаження здійснювався моніторинг поточної завантаженості обчислювальних вузлів за допомогою утиліти *vmstat*. У результаті було отримано криві завантаженості для кожного вузла (Рисунок 4.10).

Таблиця 4.5 Результати експериментальної перевірки запропонованого комплексного методу

Показник / Режим роботи методу	**Без вимкнення серверів**	**З вимкненням серверів**
Сумарне енергоспоживання системи	7619,94	7570,18
Сумарна кількість обчислювальних робіт в експерименті	235	235
Приведена сумарна кількість обчислювальних робіт в експерименті	9428,116	9428,116
Кількість втрачених обчислювальних робіт	6	6
Приведена кількість втрачених обчислювальних робіт	235,211	235,211
Приведена кількість успішно оброблених обчислювальних робіт ω'	9192,905	9192,905
Коефіцієнт готовності системи p_{avail}	**0,9751**	**0,9751**
Показник продуктивності $C_\Sigma = \omega'/T$ [робіт/год]	**383,038**	**383,038**
Показник енергоефективності $E_\Sigma = \frac{\omega'}{\sum_{j=1}^{n} W_j}$ [робіт/Вт*год]	**1,2064**	**1,2144**
Критерій ефективності $K_{opt} = E_\Sigma * C_\Sigma \to max$, при $p_{avail} \geq p_{avail_{SLA}}$	**462,097 при 0,9751 > 0,95**	**465,161 при 0,9751 > 0,95**

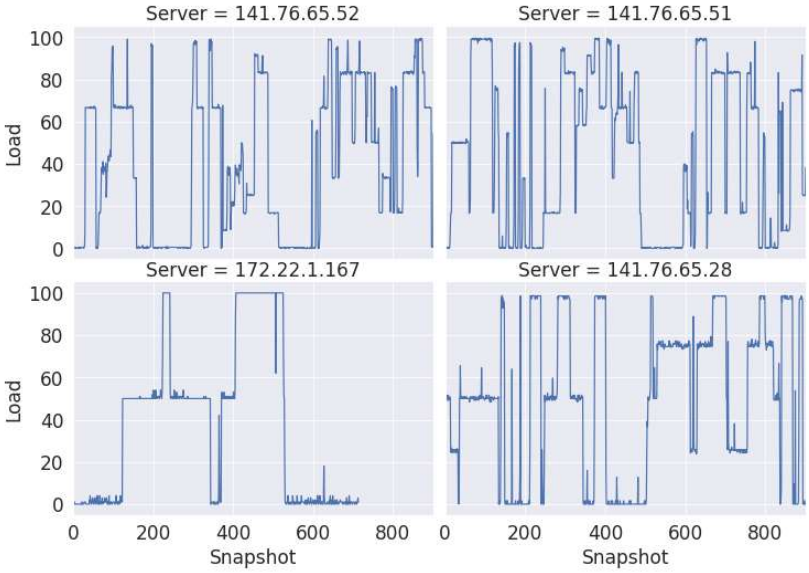


Рисунок 4.10 Криві поточної завантаженості вузлів при використанні алгоритму планування Backfill

Варто зазначити, що у порівнянні із запропонованим комплексним методом, при використанні алгоритму планування Backfill завантаженість кожного з вузлів є менш рівномірною. Це обумовлено тим, що у моменти низької інтенсивності вхідного навантаження усі вузли обробки знаходяться в активному стані і готові приймати обчислювальні роботи на обробку. У цій ситуації “збитковості” ресурсів їх частина так чи інакше знаходиться у стані простою, що не є вигідним з точки зору енергоспоживання.

Так само як і при проведенні оцінки запропонованого комплексного методу, на основі даних про поточну завантаженість вузлів та моделей енергоспоживання вузлів за формулою (4.2) було визначено сумарне енергоспоживання кожного вузла та системи в цілому. Результати визначення енергоспоживання наведено на рисунку (Рисунок 4.11).

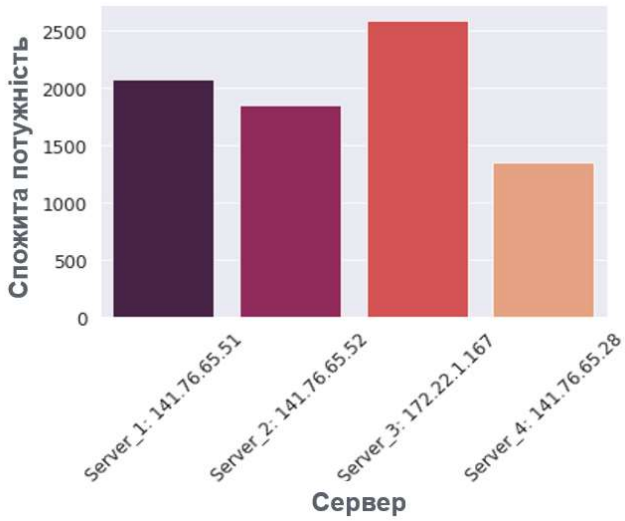


Рисунок 4.11 Порівняння сумарного енергоспоживання вузлів в ході експериментальної оцінки підходу Backfill

Як видно з графіка на рисунку, найменш ефективний сервер “Server_3: 172.22.1.167” спожив суттєвий обсяг енергії, оскільки не дивлячись на свою невисоку ефективність використовувався у повній мірі для обслуговування навантаження. Для порівняння, у рамках запропонованого комплексного методу (Рисунок 4.9) цей сервер використовувався у значно меншій мірі та як наслідок спожив менше електроенергії.

У таблиці (Таблиця 4.6) наведено результати експериментальної перевірки підходу Backfill.

Як видно з таблиці, у випадку використання алгоритму планування навантаження Backfill усі задачі були успішно оброблені. Це обумовлено тим, що кількість ресурсів у системі у кожний момент часу була більш ніж достатньою, оскільки переведення серверів у режим сну не використовувалось. Однак, при цьому системою було спожито більше електроенергії, оскільки усі вузли значну частину часу працювали у режимі недостатньої завантаженості.

Таблиця 4.6 Результати експериментальної перевірки підходу Backfill

Показник	**Значення для підходу Backfill**
Сумарне енергоспоживання системи	7856,67
Сумарна кількість обчислювальних робіт в експерименті	222
Приведена сумарна кількість обчислювальних робіт в експерименті	8706,01
Кількість втрачених обчислювальних робіт	0
Приведена кількість успішно оброблених обчислювальних робіт ω'	8706,01
Коефіцієнт готовності системи p_{avail}	**1**
Показник продуктивності $C_{\Sigma} = \omega'/T$ [$\frac{\text{робіт}}{\text{год}}$]	**362,750**
Показник енергоефективності $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j}$ [$\frac{\text{робіт}}{\text{Вт*год}}$]	**1,1081**
Критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max$, при $p_{avail} \geq p_{avail_{SLA}}$	**401,963 при 1 > 0,95**

4.3.1.2 Оцінка ефективності алгоритму планування Round Robin (порівняння із запропонованим комплексним методом)

Аналогічним чином було виконано експериментальну оцінку роботи алгоритму планування навантаження Round Robin та його порівняння із запропонованим комплексним методом. На рисунку (Рисунок 4.12) зображено криві завантаженості для кожного обчислювального вузла отримані за допомогою утиліти *vmstat*.

Рисунок 4.12 Криві поточної завантаженості вузлів при використанні алгоритму планування Round Robin

Як видно з рисунка, у випадку використання Round Robin вузли завантажені нерівномірно, більшу частину часу вони знаходяться у стані низької завантаженості і при цьому неефективно споживають електроенергію. На рисунку (Рисунок 4.13) наведено результати щодо енергоспоживання обчислювальних вузлів при використанні Round Robin, отримані за допомогою формули (4.2), а результати експерименту щодо показників ефективності системи в цілому наведено у таблиці (Таблиця 4.7).

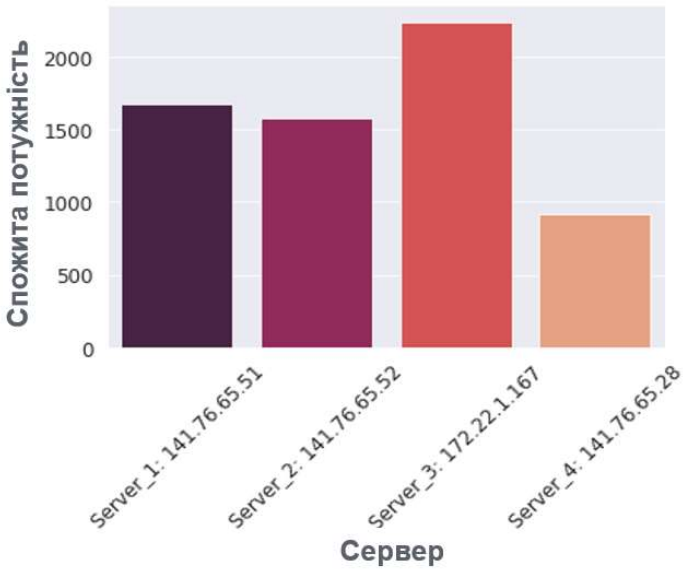


Рисунок 4.13 Порівняння сумарного енергоспоживання вузлів в ході експериментальної оцінки підходу Round Robin

Як видно з рисунка, як і у випадку з використанням підходу Backfill, при використанні Round Robin значний внесок у сумарне споживання електроенергії вніс малоефективний вузол “Server_3: 172.22.1.167”.

Таблиця 4.7 Результати експериментальної перевірки підходу Round Robin

Показник	**Значення для підходу Round Robin**
Сумарне енергоспоживання системи	6400,49
Сумарна кількість обчислювальних робіт в експерименті	235
Приведена сумарна кількість обчислювальних робіт в експерименті	7424,764
Кількість втрачених обчислювальних робіт	40
Приведена кількість втрачених обчислювальних робіт	1273,32
Приведена кількість успішно оброблених обчислювальних робіт ω'	6150,832
Коефіцієнт готовності системи p_{avail}	**0,8284**
Показник продуктивності $C_{\Sigma} = \omega'/T$ [$\frac{\text{робіт}}{\text{год}}$]	**256,285**
Показник енергоефективності $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j}$ [$\frac{\text{робіт}}{\text{Вт*год}}$]	**0,9609**
Критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max$, при $p_{avail} \geq p_{avail_{SLA}}$	**246,264 при 0,8284 < 0,95 – порушення критерію**

4.3.1.3 Аналіз результатів експериментальної оцінки ефективності запропонованого комплексного методу та його порівняння з існуючими підходами

Проведемо порівняння запропонованого комплексного методу із підходами Backfill та Round Robin. Для цього наведемо усі отримані дані щодо оцінюваних показників ефективності у таблиці (Таблиця 4.8).

Таблиця 4.8 Аналіз результатів експериментальної оцінки, порівняння запропонованого комплексного методу з існуючими підходами

Підхід / Показник	**Round Robin**	**Backfill**	**Запропонований підхід без вимкнення серверів**	**Запропонований підхід з вимкненням серверів**
Сумарне енергоспоживання системи	6400,49	7856,67	7619,94	7570,18
Коефіцієнт готовності системи p_{avail}	0,8284	1	0,9751	0,9751
Відповідність вимогам SLA щодо коефіцієнта готовності: $p_{avail} \geq p_{avail_{SLA}}$	0,8284 < 0,95 - не відповідає	1 > 0,95 - відповідає	0,9751 > 0,95 - відповідає	0,9751 > 0,95 - відповідає
Показник продуктивності $C_{\Sigma} = \omega'/T$ [робіт/год]	256,285	362,750	383,038	383,038
Показник енергоефективності $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j}$ [робіт/Вт*год]	0,9609	1,1081	1,2064	1,2144
Критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max$, при $p_{avail} \geq p_{avail_{SLA}}$	246,264 при 0,8284 < 0,95 – порушення критерію	401,963 при 1 > 0,95	462,097 при 0,9751 > 0,95	465,161 при 0,9751 > 0,95
Виграш за показником енергоефективності	-	-	**8,871 %** порівняно з Backfill **25,549 %** порівняно з Round Robin	**9,953 %** порівняно з Backfill **26,382 %** порівняно з Round Robin
Виграш за показником продуктивності	-	-	-	**5,593 %** порівняно з Backfill **49,458 %** порівняно з Round Robin

Таблиця 4.8 (продовження)

Підхід / Показник	Round Robin	Backfill	Запропонований підхід без вимкнення серверів	Запропонований підхід з вимкненням серверів
Виграш за значенням коефіцієнта готовності (показник доступності)	-	-	-	**-2,49 %** порівняно з Backfill **17,709 %** порівняно з Round Robin
Виграш за критерієм ефективності	-	-	-	**15,722 %** порівняно з Backfill **88,887 %** порівняно з Round Robin при дотриманні вимог щодо доступності системи

Як видно з таблиці, запропонований комплексний метод дозволяє отримати до 9,953% виграшу за показником енергоефективності у порівнянні з відомим підходом до енергоефективного планування навантаження Backfill. У порівнянні із Round Robin виграш становить 26,382%. Це обумовлено тим, що на відміну від існуючих підходів, запропонований комплексний метод враховує індивідуальні характеристики енергоспоживання обчислювальних вузлів, комбінує переваги енергоефективного планування задач та підходів масштабування, забезпечує виконання вимог до якості сервісу шляхом побудови шаблонів масштабування із урахуванням можливих динамічних змін навантаження.

При цьому, виграш запропонованого комплексного методу за показником продуктивності становить 5,593% та 49,458% у порівнянні із Backfill та Round Robin відповідно. Виграш за показником продуктивності обумовлений в першу чергу тим, що завдяки атестації та сортуванню вузлів для обслуговування навантаження здебільшого використовуються найбільш продуктивні та

енергоефективні вузли. Ці вузли завантажуються якомога щільніше, що підвищує продуктивність обробки при менших витратах енергії.

Щодо коефіцієнта готовності системи – запропонований комплексний метод дещо програє за цим параметром енергоефективному підходу Backfill. Програш становить близько 2,49%.

За визначеним критерієм ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avai\ SLA}$ запропонований комплексний метод демонструє приріст ефективності на 15,722% у порівнянні із Backfill та на 88,887% у порівнянні з Round Robin. При цьому виконується вимога $p_{avail} \geq p_{avail_{SLA}}$, хоча абсолютне значення коефіцієнта доступності системи при використанні запропонованого комплексного методу у порівнянні із Backfill є гіршим. Це обумовлено тим, що на відміну від запропонованого комплексного методу Backfill “має у розпорядженні” усі обчислювальні вузли у кожен момент часу. Цей програш можна вважати допустимим, оскільки коефіцієнт готовності системи для запропонованого методу мав допустиме значення у відповідності до вимог SLA.

Таким чином, експериментальна оцінка запропонованого комплексного методу показала, що у досліджуваному кластері він дозволяє досягти виграшу за обраним критерієм ефективності без порушення вимог щодо доступності системи.

4.3.2 Імітаційне моделювання процесу обслуговування навантаження

З метою оцінки ефективності запропонованого комплексного методу у більших за розміром серверних кластерах та розподілених ЦОД в цілому, було проведено імітаційне моделювання роботи методу. Імітаційну модель створено з використанням мови програмування Python з використанням бібліотеки багатопоточного програмування *threading* [136] та засобів аналізу та візуалізації даних *pandas* [137] та *seaborn* [138].

Імітаційне моделювання було проведено за таким планом:

1. Первинний збір інформації та постановка задачі моделювання;
2. Створення моделі;

3. Перевірка адекватності моделі шляхом моделювання процесу обробки вхідного навантаження в умовах аналогічних лабораторному експерименту. Порівняння результатів моделювання та лабораторного експерименту;

4. Проведення моделювання для серверного кластера з 20 вузлів та обробка отриманих результатів;

5. Аналіз результатів моделювання.

4.3.2.1 Первинний збір інформації та постановка задачі моделювання

Метою первинного збору інформації та постановки задачі моделювання є отримання вхідних даних для моделі та визначення оцінюваних вихідних даних.

В рамках імітаційного моделювання було розглянуто систему з 20 обчислювальних вузлів, що працюють як єдиний обчислювальний кластер, організований згідно з топологією “зірка”.

Як видно із побудованої у пункті 2.1.2 онтологічної моделі, необхідними вхідними даними для моделі є:

1. Характеристики вхідного робочого навантаження:

– статистична крива навантаження;

– вимоги обчислювальних робіт.

2. Характеристики обчислювальних вузлів:

– апаратні характеристики;

– характеристики програмного забезпечення (у даному випадку ним виступає описане у підрозділі 4.1 ПЗ керування обчислювальними ресурсами).

Оцінюваними вихідними показниками (як і для лабораторного експерименту) при цьому є:

1. Показник енергоефективності: $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j} [\frac{\text{робіт}}{\text{Вт*год}}]$, де ω' – приведена кількість успішно оброблених обчислювальних робіт; W_j – сумарний обсяг електроенергії спожитої j-им вузлом; n – кількість активних вузлів у системі.

2. Показник продуктивності: $C_{\Sigma} = \frac{\omega'}{T} [\frac{\text{робіт}}{\text{год}}]$, де ω'– приведена кількість

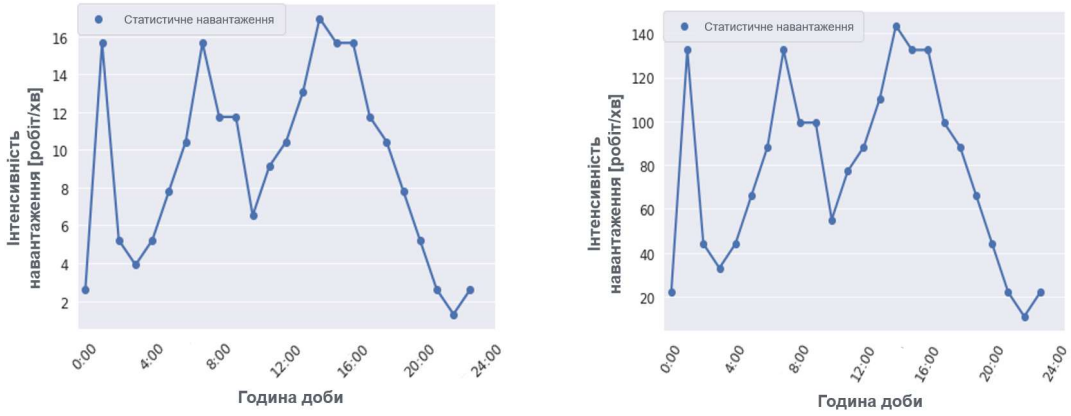
успішно оброблених обчислювальних робіт; T – тривалість обробки.

3. Коефіцієнт готовності системи: $p_{avail} = 1 - p_{loss}$, де p_{loss} – імовірність втрати запиту.

та критерій ефективності: $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$.

Для моделювання вхідного навантаження було використано ту саму криву інтенсивності вхідного навантаження, що і для лабораторного експерименту (Рисунок 4.14 (а)), величини інтенсивності в точках якої розраховані на основі даних з джерела [135] за формулою (2.8).

При моделюванні серверного кластера з 20 вузлів абсолютні значення інтенсивності було пропорційно збільшено з огляду на збільшення розміру серверного кластера (коефіцієнт пропорційності визначено на основі спостережень за ступенем відповідності поведінки моделі лабораторному експерименту) (Рисунок 4.14 (б)). По осі абсцис відкладено час у хвилинах, по осі ординат – кількість запитів (робіт, що потребують обробки) за одиницю часу.



а) б)

Рисунок 4.14 Статистична крива навантаження, використана при моделюванні побудована на основі статистики навантаження на ЦОД [135]

а) для кластера з 4 вузлів (перевірка адекватності моделі); б) для кластера з 20 вузлів.

При моделюванні обчислювальних вузлів було використано:

– ***При перевірці адекватності моделі***: Характеристики серверного обладнання з лабораторного експерименту;

– ***При моделюванні для кластера з 20 вузлів***: Характеристики серверного обладнання щодо кількості ядер (Таблиця 4.9), номінальної продуктивності (Таблиця 4.9) та моделей енергоспоживання (Рисунок 4.15).

Для порівняння із запропонованим комплексним методом, як і для випадку лабораторного експерименту, було використано:

– енергоефективний підхід щодо розподілу навантаження Backfill без використання горизонтального масштабування;

– підхід щодо розподілу навантаження Round Robin.

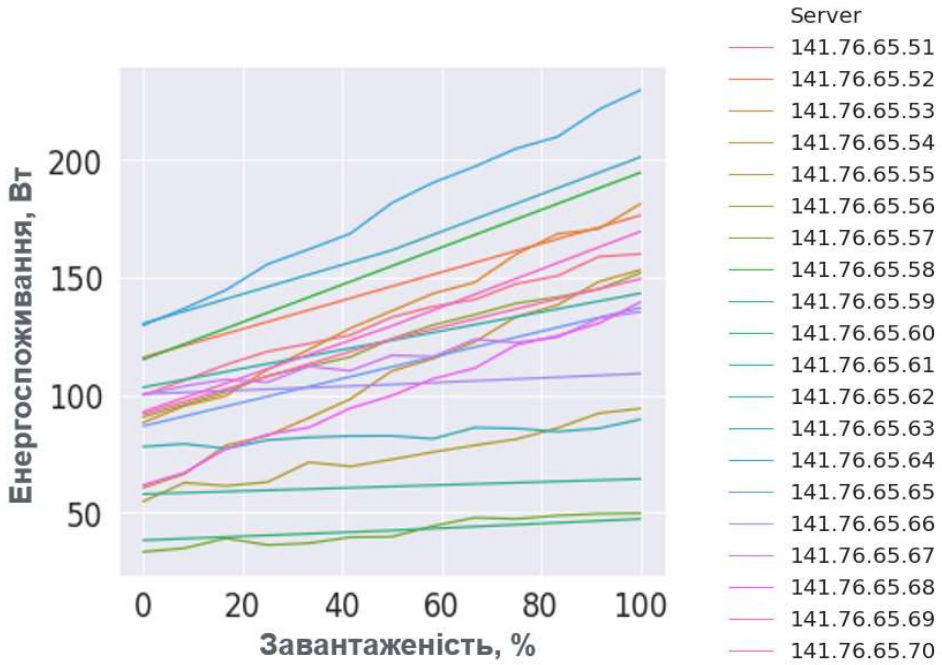


Рисунок 4.15 Моделі енергоспоживання обчислювальних вузлів використані при імітаційному моделюванні

Таблиця 4.9 Характеристики серверного обладнання використані в моделі

Сервер (IP-адреса)	**Кількість обчислювальних ядер**	**Номінальна продуктивність (у відносних одиницях Pt)**
141.76.65.51	12	8742,816
141.76.65.52	8	8732,869
41.76.65.53	2	7555,706
141.76.65.54	4	13312,358
141.76.65.55	8	8741,893
141.76.65.56	12	7504,914
141.76.65.57	2	13342,508
141.76.65.58	2	8712,984
141.76.65.59	8	7536,523
141.76.65.60	12	13306,983
141.76.65.61	4	8749,769
141.76.65.62	4	7515,126
141.76.65.63	8	13290,740
141.76.65.64	8	8680,375
141.76.65.65	2	7481,961
141.76.65.66	2	13355,885
141.76.65.67	12	8761,749
141.76.65.68	12	7474,878
141.76.65.69	2	13334,395
141.76.65.70	8	8715,443

Визначимо додаткові вхідні дані для моделювання:

1. Довжина загальної черги в системі – $Q = 5$ обчислювальних робіт (що відповідає також кількості заявок СМО);

2. Середня завантаженість процесора вузлів, яку забезпечує Backfill – $CPU_{av\ \ j} = 75\%$ [125];

3. Середня інтенсивність обробки заявок для одного вузла – $\mu'_{avg} = 0{,}03$ запити на секунду;

4. Максимально допустима імовірність втрати заявки, згідно з SLA – $p_{loss_{SLA}} = 5 * 10^{-2}$;

5. Середня кількість обчислювальних ядер одного вузла – 7. Тому, у термінах СМО, система має $n' = 7 * n = 140$ пристроїв обробки СМО.

Вхідні дані для імітаційної моделі наведено у таблиці (Таблиця 4.10).

Таблиця 4.10 Вхідні дані для імітаційної моделі

Параметр	**Опис**
$n = 20$	Кількість обчислювальних вузлів у кластері
$n' = 7 * n = 140$	Кількість пристроїв (каналів) у СМО (при середній кількості ядер для кожного вузла – 7)
$P_j = f(CPU_j)$	Експериментально визначені функції енергоспоживання (Рисунок 4.15) (для кожного j-го вузла)
C_j	Номінальна продуктивність обчислювальних вузлів (для кожного j-го вузла) (Таблиця 4.9)
$Q = 5$	Довжина черги [обчислювальних робіт]
$CPU_{max_j} = 75\%$	Середній максимальний рівень завантаженості процесора кожного j-го вузла при використанні алгоритма розподілу навантаження Backfill [125]
$\mu'_{avg} = 0{,}03$	Середня інтенсивність обробки [обчислювальних робіт на секунду].
$p_{loss_{SLA}} = 5 * 10^{-2}$	Максимально допустима імовірність втрати запиту згідно з SLA

<u>Задачі моделювання:</u>

1. Визначення показників ефективності (енергоефективності, продуктивності та доступності системи) у випадку використання запропонованого комплексного методу у серверному кластері з 20 вузлів.

2. Порівняння ефективності запропонованого комплексного методу з існуючими підходами Backfill та Round Robin.

4.3.2.2 Створення моделі

Для створеної імітаційної моделі співвідношення реального та модельного часу становило 1:60, тобто одна година реального часу відповідала 1 хвилині модельного часу. Таким чином, моделювання процесу обслуговування навантаження протягом доби відповідало моделюванню протягом 24 хвилин.

Обчислювальні роботи, як одиниці робочого навантаження, змодельовано у вигляді дискретних подій, момент настання яких є випадковою величиною. Для моделювання цієї випадкової величини використано такий підхід: за умови відомої поточної інтенсивності вхідного навантаження (наприклад x робіт за 60 секунд) ця інтенсивність ставиться у відповідність дробовому числу $x/60$ від 0 до 1. Після чого щосекунди за допомогою генератора випадкових чисел генерується число від 0 до 1 згідно з рівномірним розподілом. Якщо згенероване число менше $x/60$ така подія вважається успіхом (прихід задачі). Якщо ні – вважається, що цієї секунди задача не надійшла до системи. У випадку, якщо інтенсивність надходження задач є більшою, ніж 1 задача на секунду для розрахунків використовується коротший період часу відповідно.

Кожна подія, яка відповідає надходженню задачі, при цьому характеризується мета-описом вигляду:

– $\Delta t_{max_i} = const[c]$ – час виконання i-ї роботи;

– RAM_{min_i}[ГБ] – мінімально необхідний об’єм оперативної пам’яті;

– $cores_{min_i}$[ядер] – мінімально необхідна кількість ядер процесора;

– $storage_{min_i}$[ГБ] – мінімально необхідний об’єм постійного сховища даних.

Обчислювальні вузли змодельовано у вигляді списків з k елементів, кожен з яких представляє одне обчислювальне ядро. Кожен елемент може мати значення $true$ або $false$, що відповідає вільному або зайнятому ядру відповідно.

Процес обробки обчислювальної роботи змодельовано шляхом встановлення статусу “зайнято” ($false$) для відповідної кількості ядер j-го обчислювального вузла на час, що відповідає параметру Δt_{max_i} (Δt_{max_i}- час виконання i-ї роботи).

4.3.2.3 Перевірка адекватності моделі шляхом моделювання процесу обробки вхідного навантаження в умовах аналогічних лабораторному експерименту

Для перевірки адекватності створеної моделі проведено моделювання роботи серверного кластера аналогічного кластеру використаному у рамках лабораторного експерименту. Результати імітаційного моделювання наведено у таблиці (Таблиця 4.11). Докладні результати наведено у додатках (Додаток 2).

Таблиця 4.11 Результати імітаційного моделювання проведеного з метою перевірки адекватності створеної моделі

Підхід / Показник	**Round Robin**	**Backfill**	**Запропонований підхід без вимкнення серверів**	**Запропонований підхід з вимкненням серверів**
Сумарне енергоспоживання системи	8488,38	9481,80	9528,72	9502,7
Сумарна кількість обчислювальних робіт в експерименті	241	234	263	263
Приведена кількість обчислювальних робіт в експерименті	8677,643	8654,042	9757,821	9757,821
Кількість втрачених обчислювальних робіт	48	0	11	11
Приведена кількість втрачених обчислювальних робіт	1932,709	0	310,269	310,269
Приведена кількість успішно оброблених обчислювальних робіт ω'	6744,934	8654,042	9447,552	9447,552
Коефіцієнт готовності системи p_{avail}	0,7773	1	0,9682	0,9682
Відповідність вимогам SLA щодо коефіцієнта готовності: $p_{avail} \geq p_{avail_{SLA}}$	0,7773 < 0,95 - не відповідає	1 > 0,95 - відповідає	0,9682 > 0,95 - відповідає	0,9682 > 0,95 - відповідає
Показник продуктивності $C_{\Sigma} = \omega'/T$ $[\frac{\text{робіт}}{\text{год}}]$	281,034	360,588	393,648	393,648

Таблиця 4.11 (продовження)

Підхід / Показник	Round Robin	Backfill	Запропонований підхід без вимкнення серверів	Запропонований підхід з вимкненням серверів
Показник енергоефективності $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j} [\frac{\text{робіт}}{\text{Вт*год}}]$	0,7946	0,9127	0,9915	0,9942
Критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$	223,310 при 0,7773 < 0,95 - не відповідає	329,109 при 1 > 0,95	390,302 при 0,9682 > 0,95	391,365 при 0,9682 > 0,95
Виграш за показником енергоефективності	-	-	**8,631 %** порівняно з Backfill **24,776 %** порівняно з Round Robin	**8,929%** порівняно з Backfill **25,117 %** порівняно з Round Robin
Виграш за показником продуктивності	-	-	-	**9,168 %** порівняно з Backfill **40,071 %** порівняно з Round Robin
Виграш за значенням коефіцієнта готовності (показник доступності)	-	-	-	**-3,18 %** порівняно з Backfill **24,559 %** порівняно з Round Robin
Виграш за критерієм ефективності				**18,917%** порівняно з Backfill **75,257%** порівняно з RR при дотриманні вимог щодо доступності

Проаналізуємо ступінь адекватності створеної імітаційної моделі реальному експерименту. Для цього порівняємо отримані значення для показників ефективності (Таблиця 4.12).

Таблиця 4.12 Порівняння значень виграшів за показниками ефективності отриманих у ході експерименту та імітаційного моделювання

Виграш	**В експерименті**		**У моделі**	
	Порівняно з Backfill	**Порівняно з Round Robin**	**Порівняно з Backfill**	**Порівняно з Round Robin**
Максимальний виграш за показником енергоефективн ості	9,953 %	26,382 %	8,929%	25,117 %
Виграш за показником продуктивності	5,593 %	49,458 %	9,168 %	40,071 %
Виграш за значенням коефіцієнта готовності (показник доступності)	-2,49 %	17,709 %	-3,18 %	24,559 %

Визначимо ступінь адекватності імітаційної моделі за допомогою критерія Фішера [139]. Критерій Фішера призначений для порівняння двох незалежних груп, розділених за факторною ознакою. Цей критерій підходить для порівняння невеликих вибірок.

Для визначення розрахункового значення критерія Фішера представимо отримані у ході експерименту виграші у вигляді функції $Y = f(x)$, де x- це тип оцінюваного показника (наприклад “показник енергоефективності у порівнянні з Backfill”). Таким чином отримуємо 6 точок функції. У свою чергу, виграші отримані у ході моделювання позначимо функцією $Y_{\text{розрах}} = f(x)$ того ж аргумента. За допомогою критерія Фішера порівняємо ці дві функції. Для цього необхідно визначити розрахункове значення критерія за формулою:

$$F_{\text{розрах}} = \frac{\sum_{i=1}^{N}(Y_{i,\text{розрах}} - \bar{Y}_{\text{розрах}})^2}{m} * \frac{n - m - 1}{\sum_{i=1}^{N}(Y_i - Y_{i,\text{розрах}})^2}, \quad (4.3)$$

де $Y_{i,\text{розрах}}$ – значення відповідного отриманого виграшу;

$\bar{Y}_{\text{розрах}}$ – середнє значення отриманого виграшу;

m – кількість залежних змінних;

n – кількість вимірювань (точок, що порівнюються);

та порівняти отримане значення із табличним для визначеної достовірності. Якщо $F_{\text{розрах}} > F_{\text{табл}}$ модель можна вважати адекватною із заданою достовірністю. Приймемо достовірність рівною 95%.

За заданих умов, розрахунок матиме вигляд:

$$F_{\text{розрах}} = \frac{(8{,}929 - 17{,}444)^2 + (9{,}168 - 17{,}444)^2 + (-3{,}18 - 17{,}444)^2 + (25{,}117 - 17{,}444)^2 + (40{,}071 - 17{,}444)^2 + (24{,}559 - 17{,}444)^2}{1} \times$$
$$\times \frac{6 - 1 - 1}{(9{,}953 - 8{,}929)^2 + (5{,}593 - 9{,}168)^2 + (-2{,}49 + 3{,}18)^2 + (26{,}382 - 25{,}117)^2 + (49{,}458 - 40{,}071)^2 + (17{,}709 - 24{,}559)^2} =$$
$$= 31{,}477$$

Отримане значення порівняємо з табличним значенням $F_{\text{табл}}$=7,71 для достовірності 95%: $31{,}477 > 7{,}71$.

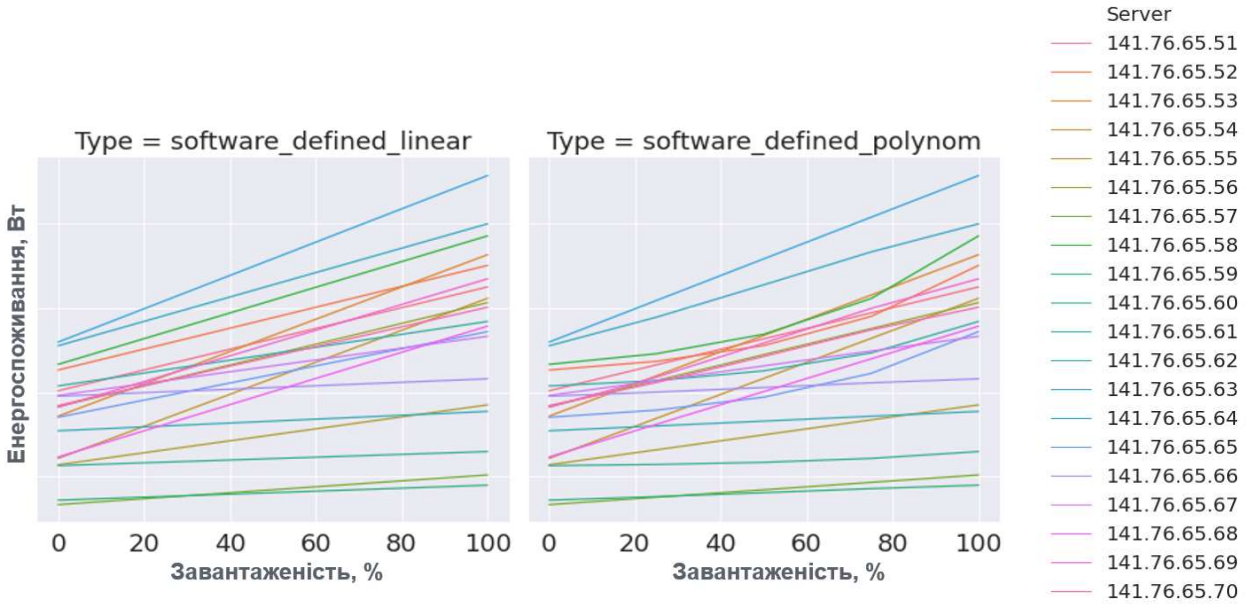
Таким чином, створену імітаційну модель можна вважати адекватною реальному об’єкту дослідження.

4.3.2.4 Проведення моделювання для серверного кластера з 20 вузлів та обробка отриманих результатів

Проведемо імітаційне моделювання для кластера з 20 вузлів згідно з вхідними даними визначеними у підпункті 4.3.2.1 .

<u>Моделювання роботи запропонованого комплексного методу</u>

Першим кроком моделювання є визначення залежностей енергоспоживання обчислювальних вузлів від завантаженості. На основі табличних даних (Додаток 1) побудовано графіки аналітичних функцій отриманих за допомогою лінійної апроксимації (Рисунок 4.16 (а)) та інтерполяції поліномом (Рисунок 4.16 (б)).



а) б)

Рисунок 4.16 Моделі енергоспоживання вузлів кластера визначені програмним методом на основі табличних даних (Додаток 1)

а) лінійна апроксимація; б) інтерполяція поліномом

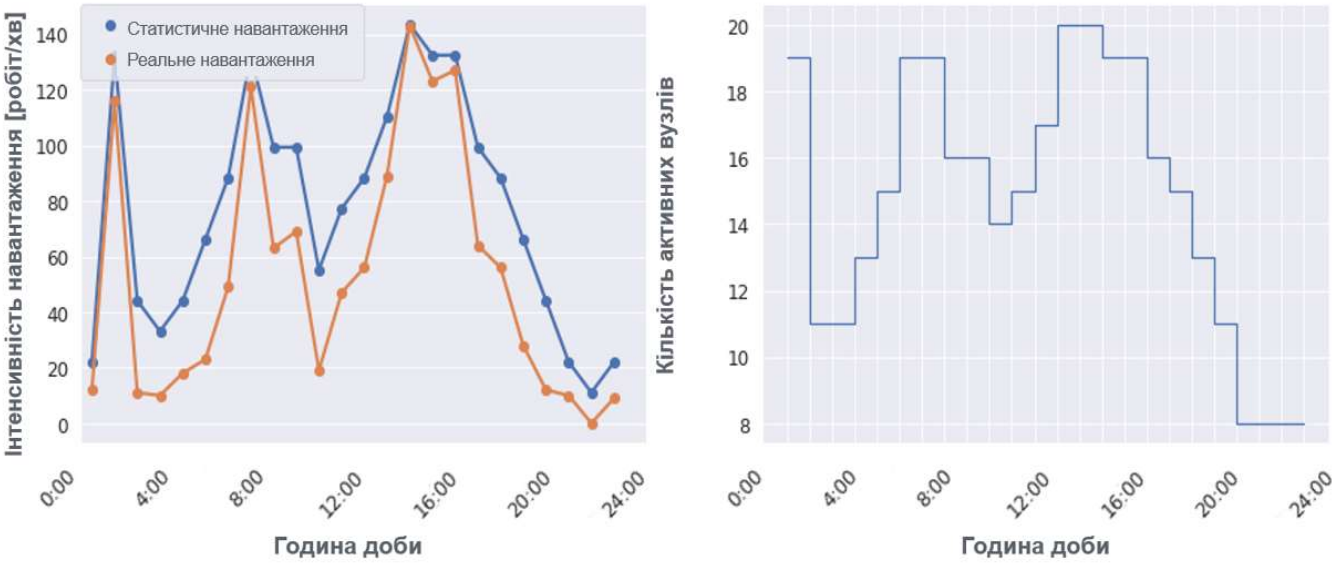
Згідно з запропонованим комплексним підходом, наступним кроком є сортування вузлів від найкращого до найгіршого за показниками продуктивності та енергоефективності. Результат сортування наведено на рисунку (Рисунок 4.17).

	Server	Total_mark	Energy_mark	Performance_mark
0	natgvo@141.76.65.67	31	17	14
1	natgvo@141.76.65.60	27	11	16
2	natgvo@141.76.65.51	27	15	12
3	natgvo@141.76.65.61	26	13	13
4	natgvo@141.76.65.57	26	7	19
5	natgvo@141.76.65.69	26	8	18
6	natgvo@141.76.65.59	25	20	5
7	natgvo@141.76.65.53	25	19	6
8	natgvo@141.76.65.66	24	4	20
9	natgvo@141.76.65.64	23	16	7
10	natgvo@141.76.65.70	23	14	9
11	natgvo@141.76.65.52	22	12	10
12	natgvo@141.76.65.62	22	18	4
13	natgvo@141.76.65.63	20	5	15
14	natgvo@141.76.65.55	20	9	11
15	natgvo@141.76.65.54	20	3	17
16	natgvo@141.76.65.58	18	10	8
17	natgvo@141.76.65.68	7	6	1
18	natgvo@141.76.65.65	4	2	2
19	natgvo@141.76.65.56	4	1	3

Рисунок 4.17 Результат сортування обчислювальних вузлів за показниками продуктивності та енергоефективності

Сортований список вузлів визначає, які вузли повинні використовуватись у процесі обробки інтенсивніше.

Наступним кроком є визначення шаблонів масштабування системи на основі даних про добову статистику навантаження. Результати визначення шаблонів та інтервалів їх дії показано на рисунку (Рисунок 4.18).

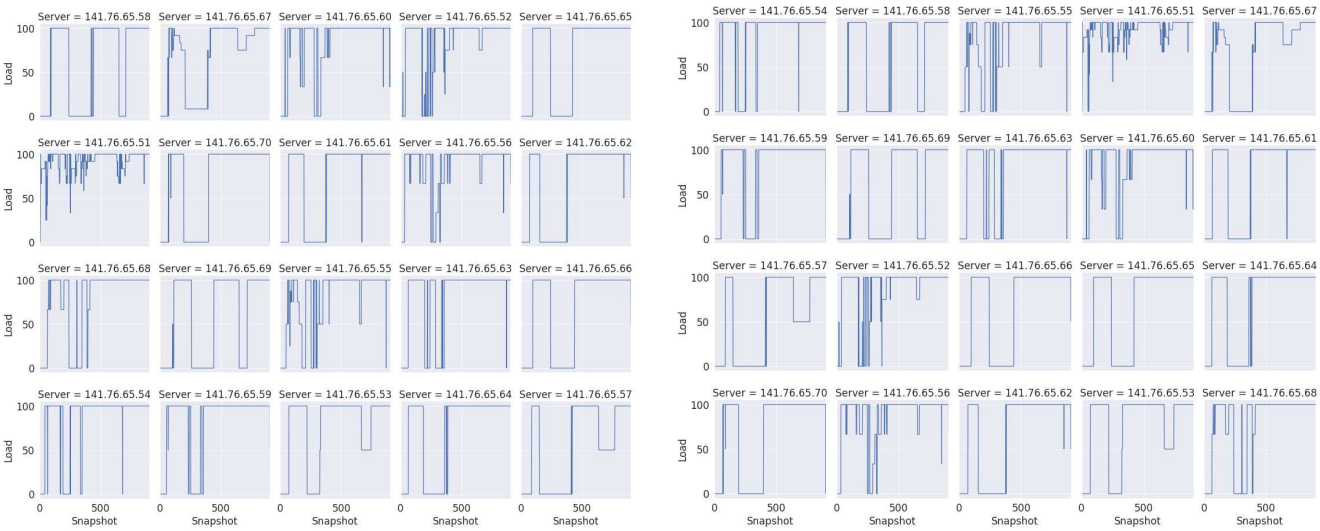


а) б)

Рисунок 4.18 Статистика інтенсивності вхідного навантаження та шаблони горизонтального масштабування визначені в експерименті

а) вхідне обчислювальне навантаження: синя крива – статистичне навантаження, помаранчева крива – реальне навантаження з відхиленнями; б) визначені шаблони масштабування

У процесі моделювання здійснювався моніторинг поточної завантаженості моделей обчислювальних вузлів (Рисунок 4.19). При цьому, на рисунку (а) наведено результати моніторингу при режимі роботи запропонованого комплексного методу, що не передбачає переведення вузлів у режим сну, а на рисунку (б) – при режимі із переведенням у режим сну відповідно.

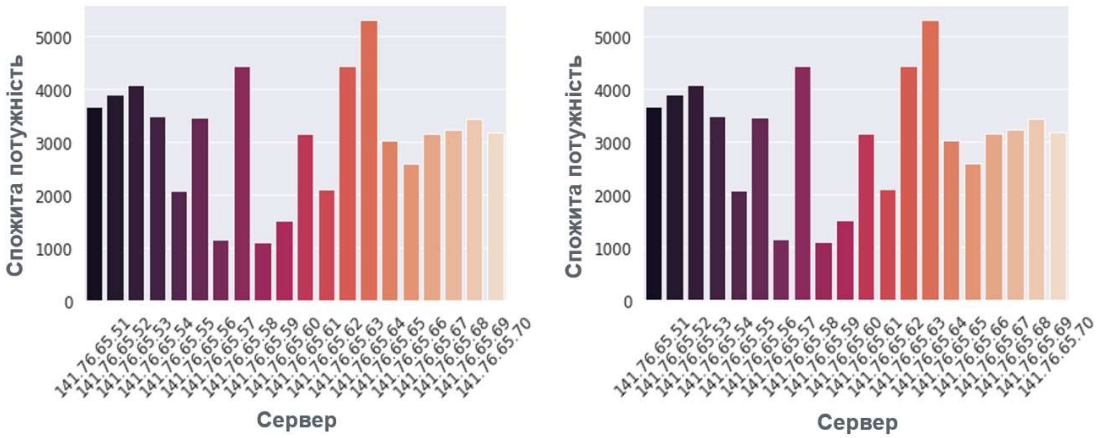


а) б)

Рисунок 4.19 Результати моніторингу поточної завантаженості вузлів**:**

а) без переведення вузлів в режим сну; б) з переведенням в режим сну

На основі даних про поточну завантаженість вузлів та функцій енергоспоживання розраховано енергоспоживання вузлів у процесі моделювання (Рисунок 4.20), а також сумарне енергоспоживання системи.



а) б)

Рисунок 4.20 Результати розрахунку енергоспоживання вузлів у процесі моделювання:

а) без переведення вузлів в режим сну; б) з переведенням в режим сну

Підсумок результатів імітаційного моделювання роботи запропонованого комплексного методу у серверному кластері з 20 вузлів наведено у таблиці (Таблиця 4.13).

Таблиця 4.13 Підсумок результатів імітаційного моделювання роботи запропонованого комплексного методу

Режим роботи методу / **Показник**	**Без вимкнення серверів**	**З вимкненням серверів**
Сумарне енергоспоживання системи	62421,85	62415,75
Сумарна кількість обчислювальних робіт в експерименті	1275	1275
Приведена сумарна кількість обчислювальних робіт в експерименті	45258,7009	45258,7009
Кількість втрачених обчислювальних робіт	37	37
Приведена кількість втрачених обчислювальних робіт	1253,1129	1253,1129
Приведена кількість успішно оброблених обчислювальних робіт ω'	44216,979	44216,979
Коефіцієнт готовності системи p_{avail}	0,97098	0,97098
Показник продуктивності $C_\Sigma = \omega'/T\ [\frac{\text{робіт}}{\text{год}}]$	1842,3742	1842,3742
Показник енергоефективності $E_\Sigma = \frac{\omega'}{\sum_{j=1}^{n} W_j}\ [\frac{\text{робіт}}{\text{Вт*год}}]$	0,70836	0,70843
Критерій ефективності $K_{opt} = E_\Sigma * C_\Sigma \to max,$ при $p_{avail} \geq p_{avail_{SLA}}$	1305,064 при 0,97098 > 0,95	1305,193 при 0,97098 > 0,95

Аналогічним чином проведено імітаційне моделювання для підходів Backfill та Round Robin. Детальні результати моделювання наведено у додатках (Додаток 3 та Додаток 4 відповідно). У таблиці (Таблиця 4.14) наведено підсумок порівняння результатів імітаційного моделювання для запропонованого комплексного методу, Backfill та Round Robin.

Таблиця 4.14 Результати імітаційного моделювання для запропонованого комплексного методу, Backfill та Round Robin у кластері з 20 вузлів

Підхід / Показник	**Round Robin**	**Backfill**	**Запропонований підхід без вимкнення серверів**	**Запропонований підхід з вимкненням серверів**
Сумарне енергоспоживання системи	54044,37	62826,63	62421,85	62415,75
Сумарна кількість обчислювальних робіт в експерименті	1294	1305	1275	1275
Приведена кількість обчислювальних робіт в експерименті	46517,723	47784,895	46185,289	46185,289
Кількість втрачених обчислювальних робіт	484	119	37	37
Приведена кількість втрачених обчислювальних робіт	17217,17	4732,768	1041,721	1041,721
Приведена кількість успішно оброблених обчислювальних робіт ω'	29300,553	43052,12699	45143,568	45143,568
Коефіцієнт готовності системи p_{avail}	0,62597	0,90881	0,97098	0,97098
Відповідність вимогам SLA щодо коефіцієнта готовності: $p_{avail} \geq p_{avail_{SLA}}$	0,62597 < 0,95- не відповідає	0,90881 < 0,95 - не відповідає	0,97098 > 0,95 - відповідає	0,97098 > 0,95 - відповідає
Показник продуктивності $C_{\Sigma} = \omega'/T$ [$\frac{\text{робіт}}{\text{год}}$]	1220,8564	1793,8386	1880,982	1880,982

Таблиця 4.14 (продовження)

Підхід / Показник	Round Robin	Backfill	Запропонований підхід без вимкнення серверів	Запропонований підхід з вимкненням серверів
Показник енергоефективності $E_{\Sigma} = \frac{\omega'}{\sum_{j=1}^{n} W_j} [\frac{\text{робіт}}{\text{Вт} * \text{год}}]$	0,54216	0,68525	0,72320	0,72327
Критерій ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$	661,8995 при 0,62597 < 0,95 - не відповідає	1229,228 при 0,90881 < 0,95 - не відповідає	1360,326 при 0,97098 > 0,95	1360,458 при 0,97098 > 0,95
Виграш за показником енергоефективності	-	-	**5,538 %** порівняно з Backfill **33,392 %** порівняно з Round Robin	**5,548 %** порівняно з Backfill **33,405 %** порівняно з Round Robin
Виграш за показником продуктивності	-	-	-	**4,858 %** порівняно з Backfill **54,070 %** порівняно з Round Robin
Виграш за значенням коефіцієнта готовності (показник доступності)	-	-	-	**6,841 %** порівняно з Backfill **55,116 %** порівняно з Round Robin
Виграш за критерієм ефективності	-	-	-	**10,676%** порівняно з Backfill **105,538%** порівняно з Round Robin при дотриманні вимог щодо доступності системи

4.3.2.5 Аналіз результатів моделювання

Проаналізуємо результати імітаційного моделювання проведеного для серверного кластера з 20 серверів. Як видно з таблиці (Таблиця 4.14), запропонований комплексний метод дозволяє отримати до 5,548% виграшу за показником енергоефективності у порівнянні з відомим підходом до енергоефективного планування навантаження Backfill. Цей виграш є дещо меншим, ніж у кластері з 4 вузлів, що при проведенні лабораторного експерименту становив 9,953%. Можливим поясненням цьому може слугувати той факт, що при збільшенні кількості вузлів у кластері зростає також і сумарна кількість електроенергії, що споживається вузлами у стані простою без обробки корисного навантаження. На жаль, досягти ситуації повної відсутності стану простою практично неможливо, оскільки для цього знадобилося б “ідеальне” вхідне навантаження, обсяг якого у кожен момент часу повністю відповідав би наявним ресурсам і займав би їх повністю. Оскільки на практиці це неможливо, частина ресурсів все одно знаходиться у стані простою і у великих кластерах результуюча частка енергії, спожитої у стані простою, виявляється більш суттєвою. У порівнянні із Round Robin виграш запропонованого комплексного методу становить 33,405%, що дещо перевищує виграш отриманий у ході лабораторного експерименту для 4 обчислювальних вузлів.

За показником продуктивності виграш запропонованого комплексного методу становить 4,858% та 54,070% у порівнянні із Backfill та Round Robin відповідно, що наближено відповідає результатам отриманим шляхом лабораторного експерименту для кластера з 4 обчислювальних вузлів.

За коефіцієнтом готовності системи запропонований комплексний метод виявився навіть дещо кращим за підхід Backfill, коефіцієнт готовності системи склав 0,97098 або 97,098%, що відповідає вимогам SLA. Можливим поясненням таких результатів у порівнянні із Backfill може слугувати той факт, що Backfill не враховує продуктивності вузлів при розподілі обчислювальних робіт, а отже може “обрати” низькопродуктивний вузол для обробки великої обчислювальної роботи і

тим самим забезпечити довше її виконання і довше зайняття відповідних ресурсів. У такій ситуації, особливо у випадку високої інтенсивності вхідного навантаження, це може стати критичним: у черзі можуть накопичуватись задачі і бути втраченими через нестачу ресурсів.

За запропонованим критерієм ефективності $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{avail_{SLA}}$, що враховує показники енергоефективності E_{Σ}, продуктивності C_{Σ} та доступності системи (коефіцієнт готовності p_{avail}) виграш запропонованого комплексного методу склав 10,676% у порівнянні із Backfill та 105,538% у порівнянні з Round Robin при дотриманні вимог щодо доступності системи. Суттєвий виграш у порівнянні з обома підходами пояснюється зокрема тим, що на відміну від них запропонований комплексний метод враховує індивідуальні характеристики енергоспоживання обчислювальних вузлів, що дозволяє обрати найбільш ефективні вузли для обслуговування навантаження. Крім того, перевага запропонованого методу полягає у застосуванні підходу масштабування, який передбачає розрахунок оптимальної кількості обчислювальних вузлів в залежності від інтенсивності вхідного навантаження та за умови дотримання вимог щодо доступності системи $p_{avail} \geq p_{avail_{SLA}}$. При цьому, у випадку динамічних відхилень інтенсивності навантаження від статистичних значень, у дію вступає механізм адаптації шаблонів масштабування, що також забезпечує виконання вимоги $p_{avail} \geq p_{avail_{SLA}}$. Суттєвий виграш у порівнянні із Round Robin забезпечується також механізмом розподілу навантаження, що враховує очікуваний час виконання кожної обчислювальної роботи Δt_{max_i} із черги, вимоги робіт щодо кількості вільних обчислювальних ядер $k_{cores_{min\,i}}$, а також поточну завантаженість кожного з вузлів CPU_j. Це дозволяє розподілити обчислювальні роботи між вузлами кластера таким чином, щоб забезпечити максимальну завантаженість вузлів CPU_j, що призводить до пришвидшення обслуговування навантаження та, як наслідок, зростання продуктивності усієї системи C_{Σ}. Крім того, при вищих рівнях завантаженості енергоефективність обчислювальних вузлів також є вищою, що дозволяє одночасно отримати виграш і за показником енергоефективності E_{Σ}.

Таким чином, імітаційне моделювання запропонованого комплексного методу показало, що у кластері з 20 вузлів він дозволяє досягти виграшу за показниками енергоефективності E_{Σ} та продуктивності C_{Σ}, а також за запропонованим критерієм ефективності системи розподіленого обслуговування навантаження $K_{opt} = E_{\Sigma} * C_{\Sigma} \rightarrow max,$ при $p_{avail} \geq p_{ava\ \ SLA}$ в інформаційно-комунікаційній мережі.

ВИСНОВКИ

1. На основі запропонованого у дисертаційній роботі комплексного методу енергоефективного обслуговування навантаження ІКМ розроблено програмне забезпечення керування обчислювальними ресурсами.

2. Описано можливості інтеграції розробленого програмного забезпечення з платформою оператора ІКМ.

3. Рекомендовано використання розробленого ПЗ в архітектурі ІКМ 5G, а саме: у розподілених ЦОД периферійної хмари для вирішення задач Edge Computing та центральної хмари для вирішення задач SDN, NFV, Network Slicing, bDDN.

4. З метою оцінки ефективності запропонованого комплексного методу розроблено імітаційну модель розподіленого ЦОД. Проведено перевірку адекватності створеної імітаційної моделі методом лабораторного експерименту на базі серверного кластера Технічного університету м. Дрездена (Німеччина).

5. Результати експериментального порівняння запропонованого комплексного методу із існуючим підходом щодо енергоефективного планування навантаження Backfill показали, що запропонований комплексний метод дозволяє досягти виграшу у 9,953% за показником енергоефективності у порівнянні із Backfill. При цьому, виграш за показником продуктивності становить 5,593% при виконанні вимог щодо допустимого значення коефіцієнта готовності системи згідно SLA. За запропонованим критерієм ефективності K_{opt} виграш склав 15,722%.

6. Експериментальне порівняння запропонованого комплексного методу із широко використовуваним підходом щодо планування навантаження Round Robin показало виграш запропонованого методу у 26,382% за показником енергоефективності та 49,458 % за показником продуктивності. За запропонованим критерієм ефективності K_{opt} виграш склав 88,887%.

7. Отримані результати основною мірою обумовлені використанням запропонованого у роботі методу горизонтального масштабування обчислювальних ресурсів із використанням індивідуальних моделей

енергоспоживання вузлів системи, а також методом адаптації шаблонів масштабування до динамічних змін інтенсивності вхідного навантаження.

8. З метою оцінки ефективності запропонованого комплексного методу у розподілених ЦОД більшого масштабу, було проведено імітаційне моделювання для серверного кластера, що складається з 20 вузлів. Результати моделювання та порівняння запропонованого комплексного методу з існуючими підходами показали перевагу запропонованого методу у 5,548% за показником енергоефективності у порівнянні з Backfill та 33,405% у порівнянні з Round Robin. За показником продуктивності виграш склав 4,858% та 54,070% відповідно при дотриманні допустимого рівня доступності системи за SLA. За запропонованим критерієм ефективності виграш запропонованого комплексного методу склав 10,676% у порівнянні із Backfill та 105,538% у порівнянні з Round Robin при дотриманні вимог щодо доступності системи.

ЗАГАЛЬНІ ВИСНОВКИ

У дисертаційній роботі розв’язано актуальну науково-практичну задачу підвищення енергоефективності та продуктивності обслуговування навантаження інформаційно-комунікаційної мережі при виконанні вимог щодо доступності системи обслуговування навантаження за рахунок застосування комплексного методу енергоефективного обслуговування навантаження.

В дисертаційній роботі отримані такі теоретичні результати:

1. Аналіз специфікацій та рекомендацій Міжнародної спілки електрозв’язку та Європейського інституту телекомунікаційних стандартів дозволив визначити основні задачі, що формують обчислювальне навантаження сучасної ІКМ. Аналіз вимог щодо обслуговування цього навантаження дозволив виділити основні показники ефективності процесу обслуговування навантаження ІКМ: енергоефективність E_{Σ} та продуктивність C_{Σ} обслуговування, а також коефіцієнт готовності системи p_{avail} як показник її доступності. На основі цих показників сформульовано критерій ефективності системи K_{opt}.

2. Аналіз існуючих підходів щодо підвищення ефективності обслуговування навантаження ІКМ (енергоефективності, продуктивності та доступності системи) показав, що відомі підходи не враховують усіх трьох зазначених показників у сукупності, що є критичним у випадку обслуговування навантаження ІКМ з огляду на необхідність гарантувати виконання вимог до якості надання послуг. Крім того, існуючі підходи не використовують індивідуальні характеристики енергоспоживання обчислювальних вузлів ЦОД, що призводить до неоптимального використання обчислювальних ресурсів.

3. Вперше запропоновано комплексний метод енергоефективного обслуговування навантаження ІКМ, що відрізняється від відомих використанням індивідуальних моделей енергоспоживання обчислювальних вузлів, поєднанням переваг підходів горизонтального масштабування та енергоефективного розподілу задач, врахуванням непередбачуваних динамічних змін інтенсивності вхідного навантаження, що є науковою новизною та дозволило підвищити

енергоефективність та продуктивність процесу обслуговування навантаження за умови дотримання вимог щодо доступності системи.

4. Набули подальшого розвитку підходи щодо формалізації процесу обслуговування навантаження в інформаційно-комунікаційній мережі шляхом побудови онтологічної моделі процесу обслуговування навантаження, що дозволило систематизувати та якісно описати складні взаємозв'язки між виділеними показниками ефективності досліджуваного процесу та факторами, що впливають на них.

5. Удосконалено математичну модель системи розподіленого обслуговування навантаження інформаційно-комунікаційної мережі як СМО. Запропонована модель відрізняється від відомих тим, що враховує змінний характер інтенсивності вхідного навантаження та можливість використання технік паралелізації при розробці ПЗ сучасних ІКМ, що є науковою новизною. Побудовану математичну модель покладено в основу нового методу розрахунку шаблонів горизонтального масштабування обчислювальної системи, що дозволило визначати оптимальну кількість активних обчислювальних вузлів у системі на кожному інтервалі часу, який визначається швидкістю зміни інтенсивності вхідного навантаження.

6. Удосконалено існуючі підходи щодо горизонтального масштабування обчислювальної системи шляхом використання індивідуальних моделей енергоспоживання обчислювальних вузлів та застосування механізму прогнозування динамічних відхилень вхідного навантаження, що є науковою новизною. Це дозволило прогнозувати та вчасно реагувати на непередбачувані зміни інтенсивності вхідного навантаження та дотримуватись вимог щодо доступності системи та продуктивності обслуговування навантаження ІКМ.

Практичні результати отримані в дисертаційній роботі:

1. На основі запропонованого комплексного методу розроблено програмне забезпечення керування обчислювальними ресурсами, що може бути використане у розподілених ЦОД периферійної та центральної хмари в інформаційно-комунікаційній мережі 5G.

2. Проведено аналіз ефективності запропонованого комплексного методу та

створеного на його основі ПЗ шляхом лабораторного експерименту та імітаційного моделювання, який показав, що виграш запропонованого комплексного методу у порівнянні із відомими підходами Backfill та Round Robin за показником енергоефективності становить 9,953% та 26,382% відповідно. Виграш за показником продуктивності становить 5,593% та 49,458% відповідно. При цьому запропонований комплексний метод забезпечує виконання вимог щодо доступності системи обслуговування та дає виграш за запропонованим критерієм ефективності на 15,722% у порівнянні із Backfill та на 88,887% у порівнянні з Round Robin.

3. Виконано впровадження запропонованого комплексного методу в компанії ТОВ «Хуавей Україна» для мереж Інтернету речей та SmartCity, що дозволило підвищити енергоефективність та продуктивність обслуговування навантаження на 14,892% за запропонованим критерієм ефективності.

Можливими напрямами продовження досліджень за тематикою дисертації є аналіз проблеми класифікації та пріоритизації навантаження ІКМ та врахування інформації отриманої у результаті класифікації та пріоритизації в процесі планування навантаження у рамках запропонованого комплексного методу, а також подальше дослідження методів та підходів щодо прогнозування відхилень інтенсивності вхідного навантаження в режимі реального часу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Masanet, E., Shehabi, A., Lei, N., Smith, S., & Koomey, J. (2020). Recalibrating global data center energy-use estimates. *Science*, 367(6481), pp. 984-986. DOI: https://doi.org/10.1126/science.aba3758.

2. Data Center Market and Technology Trends Power Electronics presentation. (2016). [Електронний ресурс] *Yole Développement, APEC*, available at: https://www.slideshare.net/Yole_Developpement/data-center-market-and-technology-trends-power-electronics-presentation-held-at-apec-2016-from-yole-dveloppement. (last accessed 18.10.2021).

3. Barroso, L. A., & Hölzle, U. (2007). The case for energy-proportional computing. *Computer*, 40(12), pp. 33-37. DOI: https://doi.org/10.1109/mc.2007.443.

4. Valentini, G. L., Lassonde, W., Khan, S. U., Min-Allah, N., Madani, S. A., Li, J., & Bouvry, P. (2013). An overview of energy efficiency techniques in cluster computing systems. *Cluster Computing*, 16(1), pp. 3-15. DOI: https://doi.org/10.1007/s10586-011-0171-x

5. Flautner, K., Reinhardt, S., & Mudge, T. (2001). Automatic performance setting for dynamic voltage scaling. *In Proceedings of the 7th annual international conference on Mobile computing and networking* (pp. 260-271).

6. Ge, R., Feng, X., & Cameron, K. W. (2005). Improvement of power-performance efficiency for high-end computing. *In 19th IEEE International Parallel and Distributed Processing Symposium* (pp. 8-pp). IEEE.

7. Ge, R., Feng, X., & Cameron, K. W. (2005). Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. *In SC'05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing* (pp. 34-34). IEEE.

8. Burd, T. D., Pering, T. A., Stratakos, A. J., & Brodersen, R. W. (2000). A dynamic voltage scaled microprocessor system. *IEEE Journal of solid-state circuits*, 35(11), pp. 1571-1580.

9. Freeh, V. W., Pan, F., Kappiah, N., Lowenthal, D. K., & Springer, R. (2005). Exploring the energy-time tradeoff in MPI programs on a power-scalable cluster. *In 19th IEEE International Parallel and Distributed Processing Symposium* (pp. 10-pp). IEEE.

10. Wang, S., Qian, Z., Yuan, J., & You, I. (2017). A DVFS based energy-efficient tasks scheduling in a data center. *IEEE Access*, 5, pp. 13090-13102. DOI: 10.1109/ACCESS.2017.2724598.

11. Lee, J., Nam, B. G., & Yoo, H. J. (2007, November). Dynamic voltage and frequency scaling (DVFS) scheme for multi-domains power management. *In 2007 IEEE Asian Solid-State Circuits Conference* (pp. 360-363). IEEE. DOI: 10.1109/ASSCC.2007.4425705.

12. Chen, G., Malkowski, K., Kandemir, M., & Raghavan, P. (2005, April). Reducing power with performance constraints for parallel sparse applications. *In 19th IEEE International Parallel and Distributed Processing Symposium* (pp. 8-13). IEEE.

13. Freeh, V. W., & Lowenthal, D. K. (2005, June). Using multiple energy gears in MPI programs on a power-scalable cluster. *In Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming* (pp. 164-173).

14. Ge, R., Feng, X., Feng, W. C., & Cameron, K. W. (2007, September). Cpu miser: A performance-directed, run-time system for power-aware clusters. *In 2007 International Conference on Parallel Processing (ICPP 2007)* (pp. 18-18). IEEE.

15. Tolentino, M. E., Turner, J., & Cameron, K. W. (2007, May). Memory-miser: a performance-constrained runtime system for power-scalable clusters. *In Proceedings of the 4th international conference on Computing frontiers* (pp. 237-246).

16. Sharma, Y., Javadi, B., Si, W., & Sun, D. (2016). Reliability and energy efficiency in cloud computing systems: Survey and taxonomy. *Journal of Network and Computer Applications*, 74, pp. 66-85. DOI: https://doi.org/10.1016/j.jnca.2016.08.010.

17. Bostoen, T., Mullender, S., & Berbers, Y. (2013). Power-reduction techniques for data-center storage systems. *ACM Computing Surveys (CSUR)*, 45(3), pp. 1-38. DOI: https://doi.org/10.1145/2480741.2480750.

18. Хританков, А. С. (2009). Модели и алгоритмы распределения нагрузки. Алгоритмы на основе сетей СМО. *Информационные технологии и вычислительные системы*, (3), сс. 33-48..

19. Beloglazov, A., & Buyya, R. (2015). OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds. *Concurrency*

and Computation: Practice and Experience, 27(5), pp. 1310-1333. DOI: https://doi.org/10.1002/cpe.3314.

20. Rybina, K., Dargie, W., Umashankar, S., & Schill, A. (2015, October). Modelling the live migration time of virtual machines. *In OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"* (pp. 575-593). Springer, Cham. DOI: 10.1007/978-3-319-26148-5_39.

21. Sudarsono, G. A., Nugroho, E. P., & Putra, R. R. J. (2019, November). Scheduling the cluster server node shutdown based on the hierarchical and k-means clustering combinations. *In Journal of Physics: Conference Series* (Vol. 1280, No. 3, p. 032039). IOP Publishing. DOI: https://doi.org/10.1088/1742-6596/1280/3/032039.

22. Rzadca, K., Findeisen, P., Swiderski, J., Zych, P., Broniek, P., Kusmierek, J., & Wilkes, J. (2020, April). Autopilot: workload autoscaling at google. *In Proceedings of the Fifteenth European Conference on Computer Systems* (pp. 1-16). DOI: 10.1145/3342195.3387524.

23. Gandhi, A., Gupta, V., Harchol-Balter, M., & Kozuch, M. A. (2010). Optimality analysis of energy-performance trade-off for server farm management. *Performance Evaluation*, 67(11), pp. 1155-1171.

24. Möbius, C., Dargie, W., & Schill, A. (2013). Power consumption estimation models for processors, virtual machines, and servers. *IEEE Transactions on Parallel and Distributed Systems*, 25(6), pp. 1600-1614.

25. Vashistha, J., & Jayswal, A. K. (2013). Comparative study of load balancing algorithms. *IOSR Journal of Engineering*, 3(3), pp. 45-50.

26. Hosseinimotlagh, S., Khunjush, F., & Hosseinimotlagh, S. (2014, February). A cooperative two-tier energy-aware scheduling for real-time tasks in computing clouds. *In 2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing* (pp. 178-182). IEEE.

27. Армента-Кано, Ф., Черных, А., Кортес-Мендоза, Х. М., Яхьяпур, Р., Дроздов, А. Ю., Буври, П., & Несмачнов, С. (2015). Min_c: стратегия неоднородной концентрации задач для энергосберегающих компьютерных расписаний. *Труды Института системного программирования РАН*, 27(6), сс. 355-380.

28. Грушин, Д. А., & Кузюрин, Н. Н. (2012). Энергоэффективные вычисления для группы кластеров. *Труды Института системного программирования РАН*, 23, сс. 433-446.

29. IBM Spectrum LSF 10.1.0 Document, (2022), [Електронний ресурс] available at: https://www.ibm.com/docs/en/spectrum-lsf/10.1.0?topic=jobs-backfill-scheduling (last accessed 04.07.2022).

30. Мазалов, В. В., & Никитина, Н. Н. (2012). Оценка характеристик алгоритма Backfill при управлении потоком задач на вычислительном кластере. *Вычислительные технологии*, 17(5), сс. 71-79.

31. Suresh, A., & Vijayakarthick, P. (2011, June). Improving scheduling of backfill algorithms using balanced spiral method for cloud metascheduler. *In 2011 International Conference on Recent Trends in Information Technology (ICRTIT)* (pp. 624-627). IEEE.

32. Lawson, B. G., Smirni, E., & Puiu, D. (2002, August). Self-adapting backfilling scheduling for parallel systems. *In Proceedings International Conference on Parallel Processing* (pp. 583-592). IEEE.

33. Liu, N., Dong, Z., & Rojas-Cessa, R. (2013, July). Task scheduling and server provisioning for energy-efficient cloud-computing data centers. *In 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops* (pp. 226-231).

34. Globa, Larysa, Gvozdetska (*Prokopets*), Nataliia & Novogrudska, Rina. (2021). Ontological model for data processing organization in information and communication networks. *System research and information technologies*. pp. 47-60. ISSN: 1681–6048. DOI: https://doi.org/10.20535/SRIT.2308-8893.2021.1.04.

35. Прокопець, Н. А., Глоба, Л. С. (2022). Математичне моделювання процесу обслуговування навантаження в інформаційно-комунікаційній мережі, *Проблеми телекомунікацій*, (1 (30)), сс. 18-31. ISSN: 2220-6922. DOI: https://doi.org/10.30837/pt.2022.1.02.

36. Prokopets, N., Globa, L. (2022). Comprehensive method of energy-efficient workload processing in the information and communication network. *Information and Telecommunication Sciences*, (2), pp. 34-45. ISSN: 2312-4121. DOI: https://doi.org/10.20535/2411-2976.22022.34-45.

37. L.Globa, O. Stryzhak, N. Gvozdetska (*Prokopets*), & V. Prokopets. (2019). Intelligent Workload Scheduling in Distributed Computing Environment for Balance Between Energy Efficiency and Performance. *Scheduling Problems - New Applications and Trends*. London, 2019. (IntechOpen). 154 c. DOI: http://dx.doi.org/10.5772/intechopen.86874

38. Globa, L., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2021). Providing Energy-efficient and High-performance Infrastructure for Smart Network. *In 2021 IEEE International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)* (pp. 133-136). IEEE., DOI: https://doi.org/10.1109/UkrMiCo52950.2021.9716620.

39. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Comprehensive energy efficient approach to workload processing in distributed computing environment. *In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)* (pp. 1-6), DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235010.

40. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Оптимізація обчислень у дата-центрі за критерієм енергоефективності. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 127-150.

41. Л.С. Глоба, Н.А. Гвоздецька (*Прокопець*), В.А. Прокопець, & О.В. Степурін (2017). Підхід до енергоефективного планування задач у серверному кластері для оптимізації обробки трафіка передачі даних. *Наукоемкие технологии в инфокоммуникациях: обработка информации, кибербезопасность, информационная борьба : розділ монографії*. Під загальною редакцією В. М. Безрука, В. В. Баранника. Х. : Издательство «Лидер», 2017. сс. 34-51.

42. Глоба, Л. С., Гвоздецька (*Прокопець*), Н. А., Прокопець, В. А., & Степурін, О. В. (2017). Енергоефективний підхід до розподілу задач у серверному кластері. *Вісник Харківського національного університету імені В.Н. Каразіна*. Серія: Математичне моделювання. Інформаційні технології. Автоматизовані

системи управління, (34), сс. 18-28. ISSN: 2524-2601.

43. Schill, A., Globa, L., Stepurin, O., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2017). Power Consumption and Performance Balance (PCPB) scheduling algorithm for computer cluster. *In 2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)* (pp. 1-8). IEEE., DOI: https://doi.org/10.1109/UkrMiCo.2017.8095365.

44. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Динамічний розподіл ресурсів у віртуалізованій мережі мобільного оператора України з підвищенням енергоефективності обробки даних. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 151-176.

45. Globa, L. S., Prokopets, V. A., & Gvozdetska (*Prokopets*), N. A. (2018). Prognostic-reactive NFV resource allocation method for implementation in virtualized mobile network EPC of Ukraine. *In 2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)* (pp. 1-5). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom.2018.8433602.

46. Globa, L., Gvozdetska (*Prokopets*), N., Prokopets, V., Stryzhak, O. (2019). QoS and Energy Efficiency Improving in Virtualized Mobile Network EPC Based on Load Balancing. In: Pejaś, J., El Fray, I., Hyla, T., Kacprzyk, J. (eds) *Advances in Soft and Hard Computing*. ACS 2018. *Advances in Intelligent Systems and Computing*, vol 889. Springer, Cham. (pp. 140-149). ISSN: 2194-5357. DOI: https://doi.org/10.1007/978-3-030-03314-9_13

47. Globa, L., Gvozdetska (*Prokopets*), N. (2019). Energy efficient workload processing in distributed computing environment modeling. *In 2019 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)* (pp. 1-6). IEEE., DOI: https://doi.org/10.1109/UkrMiCo47782.2019.9165448.

48. Globa, L., Gvozdetska (*Prokopets*), N. (2021). Experimental analysis of PCPB-2: Comprehensive energy-efficient approach to distributed workload processing in

communication networks. *In 2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)* (pp. 1-3). IEEE. DOI: https://doi.org/10.1109/BlackSeaCom52164.2021.9527759.

49. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Energy Efficient Workload Processing Technology for Distributed Computing Environment. *In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)* (pp. 1-3). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235001.

50. Gvozdetska (*Prokopets*), N., Stepurin, O., & Globa, L. (2017). Experimental analysis of PCPB scheduling algorithm. *In 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)* (pp. 244-247). IEEE., DOI: https://doi.org/10.1109/CADSM.2017.7916126.

51. Gvozdetska (*Prokopets*), N., Globa, L., & Prokopets, V. (2019). Energy-efficient Backfill-based scheduling approach for Slurm resource manager. *In 2019 IEEE 15th International Conference on the Experience of Designing and Application of CAD Systems (CADSM)* (pp. 1-5). IEEE. ISSN: 2572-7591, DOI: https://doi.org/10.1109/CADSM.2019.8779312.

52. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба. (2017). Підхід до оптимізації обробки даних з урахуванням параметрів енергоефективності та продуктивності. *Одинадцята міжнародна науково-технічна конференція "Проблеми телекомунікацій"*. 2017. сс. 464–466.

53. The law of Ukraine About telecommunications (2020), available at: https://zakon.rada.gov.ua/laws/show/1280-15#Text). (last accessed 11.10.2021).

54. View on 5G Architecture: 5G PPP Architecture Working Group, (2019), available at: https://5g-ppp.eu/wp-content/uploads/2019/07/5G-PPP-5G-Architecture-White-Paper_v3.0_PublicConsultation.pdf (last accessed 18.02.2022).

55. Європейський інститут телекомунікаційних стандартів. ETSI Standard GR NFV 003, (2020), “Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV", available at: https://www.etsi.org/deliver/etsi_gr/NFV/001_099/003/01.05.01_60/gr_NFV003v010501p.pdf (last accessed 16.09.2022).

56. Zhang, T., Qiu, H., Linguaglossa, L., Cerroni, W., & Giaccone, P. (2020). NFV platforms: Taxonomy, design choices and future challenges. *IEEE Transactions on Network and Service Management*, 18(1), pp. 30-48.

57. Міжнародна спілка електрозв'язку. ITU-T Y.3300 - Framework of software-defined networking, (2014), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-Y.3300-201406-I/en. (last accessed 20.09.2022).

58. Benzekki, K., El Fergougui, A., & Elbelrhiti Elalaoui, A. (2016). Software-defined networking (SDN): a survey. *Security and communication networks*, 9(18), pp. 5803-5833. DOI: https://doi.org/10.1002/sec.1737.

59. Khan, L. U., Yaqoob, I., Tran, N. H., Han, Z., & Hong, C. S. (2020). Network slicing: Recent advances, taxonomy, requirements, and open research challenges. *IEEE Access*, 8, pp. 36009-36028. DOI: https://doi.org/10.1109/access.2020.2975072.

60. Міжнародна спілка електрозв'язку. ITU-T Y.3156 - Framework of network slicing with AI-assisted analysis in IMT-2020 networks, (2020), [Електронний ресурс] available at: https://www.itu.int/rec/dologin_pub.asp?lang=s&id=T-REC-Y.3156-202009-I!!PDF-E&type=items. (last accessed 20.09.2022).

61. C. Bravo, H. Bäckström. (2020). Edge computing and deployment strategies for communication service providers, *Ericsson White Paper* [Електронний ресурс] available at: https://www.ericsson.com/en/digital-services/edge-computing. (last accessed 20.09.2022).

62. Ericsson | 5G for business: a 2030 market compass, (2020), [Електронний ресурс] available at: https://www.ericsson.com/assets/local/5g/the-5g-for-business-a-2030-compass-report-2019.pdf. (last accessed 20.09.2022).

63. Міжнародна спілка електрозв'язку. ITU-T F.743.12 - Requirements for edge computing in video surveillance, (2021), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-F.743.12/en. (last accessed 18.09.2022).

64. Міжнародна спілка електрозв'язку. ITU-T F.743.10 - Requirements for mobile edge computing-enabled content delivery networks, (2019), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-F.743.10. (last accessed 20.09.2022). (last accessed 20.09.2022).

65. Міжнародна спілка електрозв'язку. ITU-T Y.3652 - Big data driven networking – requirements, (2020), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-Y.3652. (last accessed 17.09.2022).

66. Cellular operators know everything about their subscribers, (2020), [Електронний ресурс] available at: https://www.vedomosti.ru/technology/articles/2015/05/26/593579 -sotovie-operatori-znayut-vse-o-svoih-abonentah. (last accessed 20.07.2021).

67. Soós, G., Ficzere, D., Varga, P., & Szalay, Z. (2020, April). Practical 5G KPI measurement results on a non-standalone architecture. In Noms 2020-2020 IEEE/IFIP network operations and management symposium (pp. 1-5). IEEE.

68. Міжнародна спілка електрозв'язку. Серія рекомендацій ITU-T L.13xx – Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant, (2022), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L. (last accessed 20.09.2022).

69. Міжнародна спілка електрозв'язку. ITU-T L.1315 - Standardization terms and trends in energy efficiency, (2017), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L.1315/en. (last accessed 20.09.2022).

70. Міжнародна спілка електрозв'язку. ITU-T L.1360 - Energy control for the software-defined networking architecture, (2016), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L.1360/en. (last accessed 20.09.2022).

71. Міжнародна спілка електрозв'язку. ITU-T L.1361 - Measurement method for energy efficiency of network functions virtualization, (2018), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L.1361. (last accessed 18.09.2022).

72. Міжнародна спілка електрозв'язку. ITU T Y.3111 - IMT-2020 network management and orchestration framework [Електронний ресурс] / Міжнародна спілка електрозв'язку. – 2017. available at: https://www.itu.int/rec/T-REC-Y.3111/en. (last accessed 18.09.2022).

73. Міжнародна спілка електрозв'язку. b-ITU-T Y.2701 - Security requirements for NGN release 1, (2017), [Електронний ресурс] available at:

https://www.itu.int/rec/T-REC-Y.2701/en. (last accessed 19.09.2022).

74. Європейський інститут телекомунікаційних стандартів. ETSI TR 132 972 - 5G; Telecommunication management; Study on system and functional aspects of energy efficiency in 5G networks (3GPP TR 32.972 version 16.1.0 Release 16), (2020), [Електронний ресурс] available at: https://www.etsi.org/deliver/etsi_tr/132900_132999/132972/16.01.00_60/tr_132972v160100p.pdf. (last accessed 19.09.2022).

75. Міжнародна спілка електрозв'язку. L.EEMDC - ITU-T work programme: Energy Efficiency in Micro Data Centre for Edge Computing, (2022), [Електронний ресурс] available at: https://www.itu.int/ITU-T/workprog/wp_item.aspx?isn=17702. (last accessed 19.09.2022).

76. Міжнародна спілка електрозв'язку. ITU-T Y.2704 - Security mechanisms and procedures for NGN, (2010), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-Y.2704/en. (last accessed 20.09.2022).

77. Міжнародна спілка електрозв'язку. ITU-T L.1302 - Assessment of energy efficiency on infrastructure in data centres and telecom centres, (2015), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L.1302. (last accessed 20.09.2022).

78. Ye, Q., Li, J., Qu, K., Zhuang, W., Shen, X. S., & Li, X. (2018). End-to-end quality of service in 5G networks: Examining the effectiveness of a network slicing framework. *IEEE Vehicular Technology Magazine*, 13(2), 65-74. DOI: https://doi.org/10.1109/mvt.2018.2809473

79. Shu, Z., & Taleb, T. (2020). A novel QoS framework for network slicing in 5G and beyond networks based on SDN and NFV. *IEEE Network*, 34(3), 256-263. DOI: https://doi.org/10.1109/mnet.001.1900423.

80. Berger, J. (2019). QoS and QoE in 5G networks Evolving applications and measurements. *In QSDG Workshop, Singapore* (pp. 19-21).

81. Васілевський, О. М. (2013). Нормування показників надійності технічних засобів : навчальний посібник. *Вінниця : ВНТУ*, 2010. – 129 с.

82. van Steen, M., & Tanenbaum, A. S. (2016). A brief introduction to distributed systems. *Computing*, 98(10), 967-1009. pp. 967–1009.

83. Таненбаум Э. (2003). Распределенные системы. Принципы и парадигмы. *Классика Computer Science*. 877 с. ISBN 5–272–00053–6.

84. Aviziens, A. (1976). Fault-tolerant systems. *IEEE transactions on computers*, 100(12), 1304-1312. DOI: 10.1109/TC.1976.1674598

85. Суперкомп'ютер (електронна енциклопеція Вікіпедія), (2021), [Електронний ресурс] available at: https://uk.wikipedia.org/wiki/Суперкомп%27ютер. (last accessed 04.10.2021).

86. Zapata, S. N. (2004). Analysis of cluster interconnection network topologies (*Doctoral dissertation, The University of Texas at El Paso*). pp. 724-729.

87. Wustenhoff, E., & BluePrints, S. (2002). Service level agreement in the data center. *Sun Microsystems Professional Series*, 2.

88. Service Availability Calculator, (2020), [Електронний ресурс] available at: https://uptime.is/99.99. (last accessed 13.05.2020).

89. QoS Class Identifier (електронна енциклопеція Вікіпедія), (2021), [Електронний ресурс] available at: https://en.wikipedia.org/wiki/QoS_Class_Identifier

90. B. George. 5G QoS: 5QI to IP DSCP Mapping : Nokia White Paper, (2020). (last accessed 20.03.2021).

91. Poola, D., Salehi, M. A., Ramamohanarao, K., & Buyya, R. (2017). A taxonomy and survey of fault-tolerant workflow management systems in cloud and distributed computing environments. *Software architecture for big data and the cloud*, 285-320. DOI: https://doi.org/10.1016/B978-0-12-805467-3.00015-6.

92. Chen, H., Wen, J., Pedrycz, W., & Wu, G. (2018). Big data processing workflows oriented real-time scheduling algorithm using task-duplication in geo-distributed clouds. *IEEE Transactions on Big Data*, 6(1), pp. 131-144.

93. A. Beloglazov, (2013). Energy-efficient management of virtual machines in data centers for cloud computing. *PhD thesis, The University of Melbourne*, 2013.

94. Wang, L., Zhan, J., Gao, W., Yang, K., Jiang, Z., Ren, R., & Luo, C. (2018). BOPS, not FLOPS! A new metric and roofline performance model for datacenter computing. *arXiv preprint arXiv:1801.09212*.

95. Рудычев, А. А., Чижова, Е. Н., Гавриловская, С. П., & Мясоедов, Р. А.

(2015). Методы принятия управленческих решений. 202 с.

96. Міжнародна спілка електрозв'язку. ITU-T L.1300 - Best practices for green data centres, (2014), [Електронний ресурс] available at: https://www.itu.int/rec/T-REC-L.1300/en. (last accessed 18.09.2022).

97. Harrington, J. L. (2016). Relational database design and implementation. *Morgan Kaufmann*.

98. Badia, A. (2005). Relational, object-oriented and object-relational data models. *In Encyclopedia of database technologies and applications* (pp. 530-535). IGI Global.

99. Nguyen, G. T., & Rieu, D. (1989). Schema evolution in object-oriented database systems. *Data & Knowledge Engineering*, 4(1), pp. 43-67. DOI: https://doi.org/10.1016/0169-023x(89)90004-9.

100. Глоба, Л. С. (2007). Математичні основи побудови інформаційно-телекомунікаційних систем: навчальний посібник для студентів спеціальності 8.092401 "Телекомунікаційні системи та мережі". *Норма-плюс*. 356 с.

101. Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology?. *In Handbook on ontologies* (pp. 1-17). Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-540-92673-3_0.

102. Globa, L., Novogrudska, R., Koval, A., & Senchenko, V. (2018). Examples of ontology model usage in engineering fields. *Ontology in Information Science*. DOI: 10.5772/intechopen.74369.

103. Bai, W. H., Xi, J. Q., Zhu, J. X., & Huang, S. W. (2015). Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model. *Mathematical Problems in Engineering*, 2015. DOI: https://doi.org/10.1155/2015/980945.

104. van der Boor, M., & Comte, C. (2021, June). Load balancing in heterogeneous server clusters: Insights from a product-form queueing model. *In 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)* (pp. 1-10). IEEE.. DOI: 10.1109/IWQOS52092.2021.9521355

105. Hu, C., Deng, Y., Yang, L.T. & Zhao, Y. (2019). Estimating the resource demand in power-aware clusters by regressing a linearly dependent relation. *IEEE*

Transactions on Sustainable Computing, 6(3), pp.385-397. DOI: 10.1109/TSUSC.2019.2894708

106. Meisner, D., & Wenisch, T. F. (2010, March). Stochastic queuing simulation for data center workloads. *In Exascale Evaluation and Research Techniques Workshop* (Vol. 16, p. 37).

107. Bukh, P. N. D. (1992). The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling. *Choice Reviews Online*. Vol. 29, no. 06. pp. 29–3347. DOI: https://doi.org/10.5860/choice.29-334.

108. Тукубаев З. Б., & Умаров А. А. (2010). Исследование потоков заявок, поступающих на веб - сервер организации. *Сборник трудов VII Молодежной школы семинара молодых ученых "Управление большими системами"*. сс. 369-373.

109. Hagihara, S., Fushihara, Y., Shimakawa, M., Tomoishi, M., & Yonezaki, N. (2017, February). Web server access trend analysis based on the Poisson distribution. *In Proceedings of the 6th International Conference on Software and Computer Applications* (pp. 256-261). DOI: 10.1145/3056662.3056701.

110. Litvinov, A.L. (2018), Theory of queuing systems: training manual, *O.M. Beketov NUUEKh, Kharkiv*, 141 p. [Литвинов, А.Л. (2018), Теорія систем масового обслуговування: навч. посібник, *ХНУМГ ім. О. М. Бекетова, Харків*, 141 с.].

111. Google Transparency Report, (2021), [Електронний ресурс] Google. Available at: https://transparencyreport.google.com/traffic/overview.

112. Wentzel, E.S. Probability Theory (4th ed.), M.: Nauka, 1969, 576 p. [Вентцель, Е.С. Теория вероятностей (4-е изд.), М.: Наука, 1969, 576 с.].

113. Padua, D. (Ed.). (2011). Encyclopedia of parallel computing. *Springer Science & Business Media*. 978 p.

114. Carvalho, G. H., Woungang, I., Anpalagan, A., & Jaseemuddin, M. (2018). Analysis of joint parallelism in wireless and cloud domains on mobile edge computing over 5G systems. *Journal of Communications and Networks*, 20(6), 565-577. DOI: 10.1109/JCN.2018.000089.

115. Badia, R. M., Conejero, J., Diaz, C., Ejarque, J., Lezzi, D., Lordan, F., &

Sirvent, R. (2015). Comp superscalar, an interoperable programming framework. *SoftwareX*, 3, 32-36.

116. Lordan Gomis, F. J., & Badia Sala, R. M. (2016). COMPSs-Mobile: parallel programming for mobile-cloud computing. *In 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2016)*: Cartagena, Colombia: 16-19 May 2016 (pp. 497-500). Institute of Electrical and Electronics Engineers (IEEE).

117. Lozhkovsky, A.G. (2012), "Queuing theory in telecommunications: textbook", *Odesa: O.S. Popov Odesa National Academy of Telecommunications*, 112 p. [Ложковский, А.Г., (2012), "Теория массового обслуживания в телекоммуникациях: підручник", *Одесса: ОНАЗ ім. ОС Попова*, 112 с.]. ISBN 978-966-7595-43-3.

118. Kremer, N.S., Putko, B.A., Trishyn, I.M., Fridman, M.N. (2017), "Research of operations in economics: a textbook for universities", *Ed. Yurayt,* Moscow, 438 p. [Кремер, Н.Ш., Путко, Б.А., Тришин, И.М., Фридман, М.Н. (2017), "Исследование операций в экономике: учебник для вузов", *Издательство Юрайт*, Москва, 438 с.]. ISBN 978-5-9916-9922-8.

119. Naumov, E.V., Berezin, V.B., Berezin, V.V., Gataulin, V.M., Monchak, A.M. (2004), "Implementation of analog-to-digital conversion of optical signals with a variable step", *Izvestia of higher educational institutions of Russia, Radioelectronics*, (3), pp.57-65. [Наумов, Е.В., Березин, В.Б., Березин, В.В., Гатаулин, В.М., Мончак, А.М. (2004), "Реализация аналого-цифрового преобразования оптических сигналов с переменным шагом", *Известия высших учебных заведений России, Радиоэлектроника*, (3), сс. 57-65.].

120. Dayarathna, M., Wen, Y., & Fan, R. (2015). Data center energy consumption modeling: A survey. IEEE Communications surveys & tutorials, 18(1), 732-794. DOI: 10.1109/COMST.2015.2481183.

121. Огородникова, О. М. (2013). Вычислительные методы в компьютерном инжиниринге: учебное пособие. *Екатеринбург : УрФУ*. 130 с.

122. Коломієць С. В. (2011). Теорія випадкових процесів: Практикум Для

студентів економічних спеціальностей вищих навчальних закладів. *Державний вищий навчальний заклад "Українська академія банківської справи Національного банку України"*, Суми, 2011. 80 с.

123. Иванова Т.В. (2017). Численные методы в оптике. Учебное пособие. *СПб: Университет ИТМО, Санкт-Петербург*, 2017. 84 с.

124. Standard Performance Evaluation Corporation, SPEC power Benchmark, (2008), [Електронний ресурс] available at: https://www.spec.org/power_ssj2008/. (last accessed 07.07.2021).

125. Mu'alem, A. W., & Feitelson, D. G. (2001). Utilization, predictability, workloads, and user runtime estimates in scheduling the IBM SP2 with backfilling. *IEEE transactions on parallel and distributed systems*, 12(6), 529-543. DOI - 10.1109/71.932708

126. Sulima, S., & Skulysh, M. (2017). Hybrid resource provisioning system for virtual network functions. *RADIO ELECTRONICS COMPUTER SCIENCE CONTROL*, (1), 16-23. DOI: 10.15588/1607-3274-2017-1-2.

127. Cui, W. (2020, September). Research and Application of Edge Computing Based on Deep Learning. *In Journal of Physics: Conference Series* (Vol. 1646, No. 1, p. 012016). IOP Publishing. DOI: 10.1088/1742-6596/1646/1/012016.

128. Stress-test, POSIX, (2022), [Електронний ресурс] available at: https://github.com/emscripten-core/posixtestsuite. (last accessed 14.01.2022).

129. CPU-stress-testing, Github, (2022), [Електронний ресурс] available at: https://github.com/ethdragon/CPU-stress-testing. (last accessed 14.01.2022).

130. 12th Marathon of Parallel Programming, (2017), [Електронний ресурс] available at: http://lspd.mackenzie.br/marathon/17/problems.html. (last accessed 14.01.2022).

131. 11th Marathon of Parallel Programming, (2016), [Електронний ресурс] available at: http://lspd.mackenzie.br/marathon/16/problems.html. (last accessed 14.01.2022).

132. Solving Every Sudoku Puzzle, (2022), [Електронний ресурс] available at: https://norvig.com/sudoku.html/. (last accessed 14.01.2022).

133. A. Holub The Game of Life. // 2004. 10.1007/978-1-4302-0725-2_3.

134. pthreads - POSIX threads. Linux manual page, (2022), [Електронний ресурс] available at: https://man7.org/linux/man-pages/man7/pthreads.7.html. (last accessed 14.01.2022).

135. Управление энергопотреблением коммерческих серверов, (2004), [Електронний ресурс] available at: https://www.osp.ru/os/2004/02/183912. (last accessed 07.11.2020).

136. threading — Thread-based parallelism, (2022), [Електронний ресурс] available at: https://docs.python.org/3/library/threading.html. (last accessed 07.07.2021).

137. Wes McKinney & the Pandas Development Team (2021) Pandas: powerful Python data analysis toolkit, [Електронний ресурс] available at: https://pandas.pydata.org/pandas-docs/version/1.4.4/pandas.pdf. (last accessed 07.07.2021).

138. Seaborn: statistical data visualization parallelism, (2022), [Електронний ресурс] available at: https://seaborn.pydata.org/. (last accessed 07.07.2021).

139. Руденко В. М. (2012). Математична статистика. Навч. посіб. *К.: Центр учбової літератури*, 2012. 304 с.

Додаток 1. Визначені функції енергоспоживання вузлів у табличному вигляді (лабораторний експеримент)

Таблиця Д1.1 Визначені функції енергоспоживання вузлів у табличному вигляді (лабораторний експеримент)

Сервер	Ступінь завантаженості процесора	Енергоспоживання
141.76.65.52	0	49,01
141.76.65.52	10	55,92
141.76.65.52	20	60,29
141.76.65.52	30	65,09
141.76.65.52	40	72,17
141.76.65.52	50	80,93
141.76.65.52	60	93,47
141.76.65.52	70	101,89
141.76.65.52	80	108,63
141.76.65.52	90	132,21
141.76.65.52	100	144,00
141.76.65.28	0	22,40
141.76.65.28	10	28,40
141.76.65.28	20	34,80
141.76.65.28	30	38,30
141.76.65.28	40	43,80
141.76.65.28	50	50,10

Сервер	Ступінь завантаженості процесора	Енергоспоживання
141.76.65.28	60	58,50
141.76.65.28	70	68,60
141.76.65.28	80	80,00
141.76.65.28	90	85,60
141.76.65.28	100	92,80
141.76.65.51	0	49,01
141.76.65.51	10	55,92
141.76.65.51	20	60,29
141.76.65.51	30	65,09
141.76.65.51	40	72,17
141.76.65.51	50	80,93
141.76.65.51	60	93,47
141.76.65.51	70	101,89
141.76.65.51	80	108,63
141.76.65.51	90	132,21
141.76.65.51	100	144,00
172.22.1.167	0	60,04
172.22.1.167	10	81,13
172.22.1.167	20	89,38

Сервер	Ступінь завантаженості процесора	Енергоспоживання
172.22.1.167	30	99,00
172.22.1.167	40	110,46
172.22.1.167	50	122,38
172.22.1.167	60	135,67
172.22.1.167	70	152,17
172.22.1.167	80	171,88
172.22.1.167	90	186,54
172.22.1.167	100	198,00

Додаток 2. Результати імітаційного моделювання з метою перевірки адекватності моделі

<u>Запропонований комплексний метод:</u>

Результати атестації:

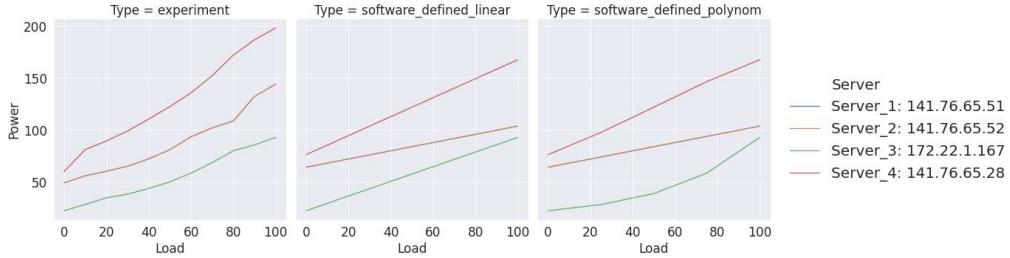


Рисунок Д2.1 Імітаційне моделювання: результати атестації вузлів

Побудовані шаблони масштабування:

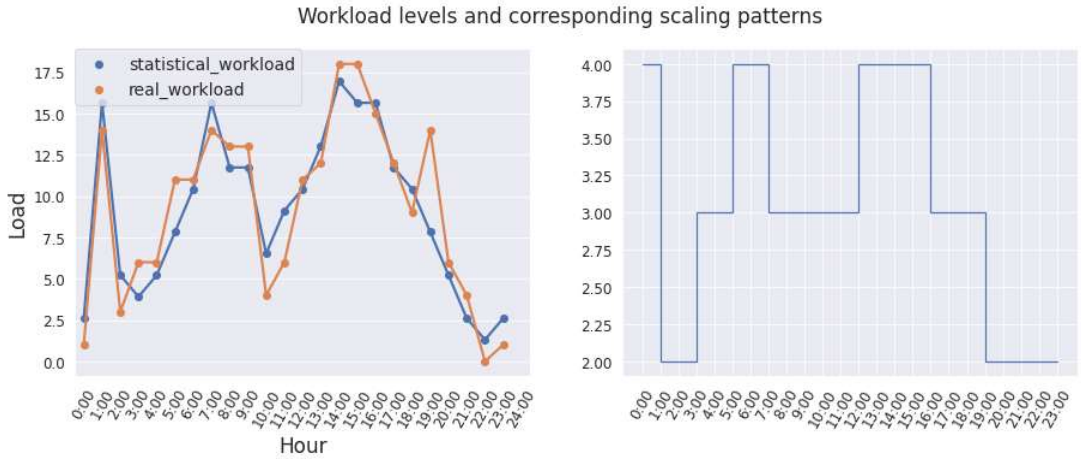


Рисунок Д2.2 Імітаційне моделювання: результати побудови шаблонів масштабування

Завантаженість серверів без переведення в режим сну та з переведенням в режим сну відповідно:

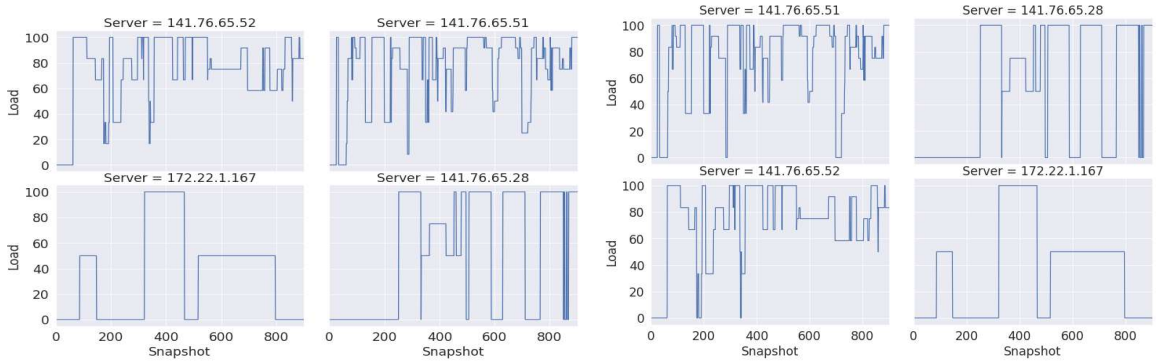


Рисунок Д2.3 Імітаційне моделювання: завантаженість серверів без переведення в режим сну та з переведенням в режим сну

Енергоспоживання серверів без переведення в режим сну та з переведенням в режим сну відповідно:

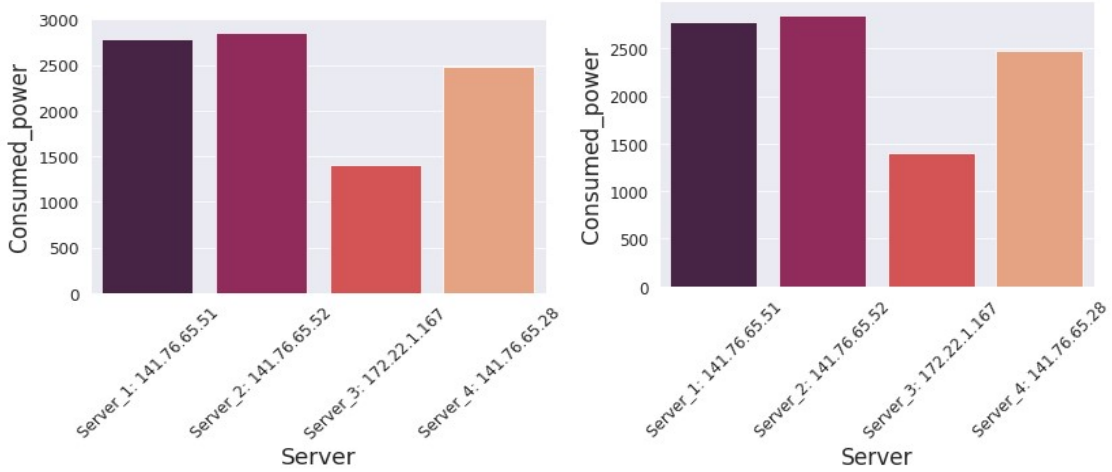


Рисунок Д2.4 Імітаційне моделювання: енергоспоживання серверів без переведення в режим сну та з переведенням в режим сну

Додаток 3. Результати імітаційного моделювання для підходу Backfill

Завантаженість серверів при використанні Backfill:

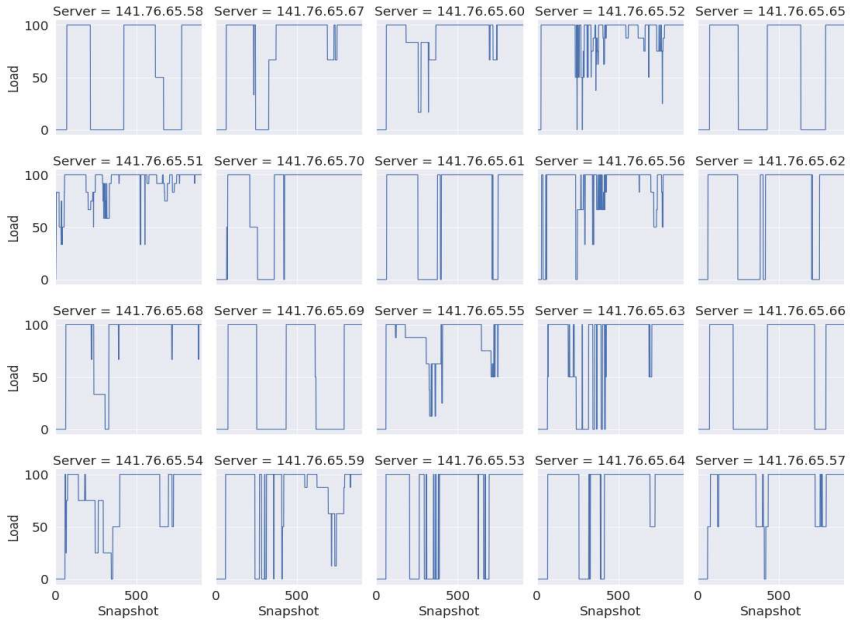


Рисунок Д3.1 Імітаційне моделювання: завантаженість серверів при використанні Backfill

Енергоспоживання серверів при використанні Backfill:

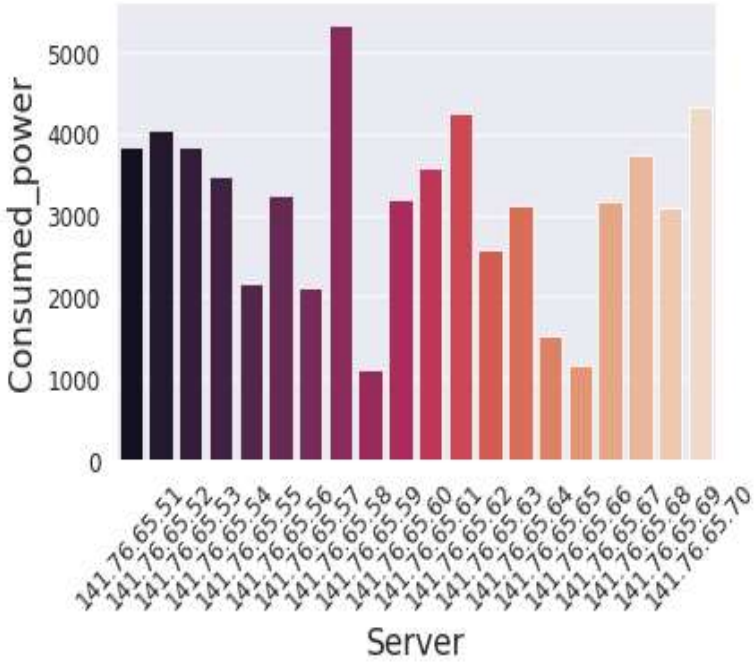


Рисунок Д3.2 Імітаційне моделювання: енергоспоживання серверів при використанні Backfill

Додаток 4. Результати імітаційного моделювання для підходу Round Robin

Завантаженість серверів при використанні Round Robin:

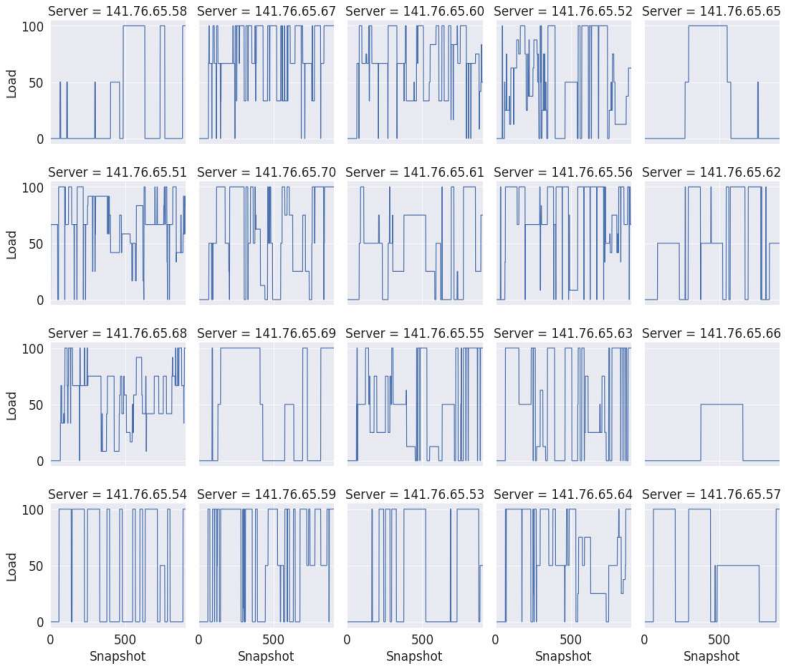


Рисунок Д4.1 Імітаційне моделювання: завантаженість серверів при використанні Round Robin

Енергоспоживання серверів при використанні Round Robin:

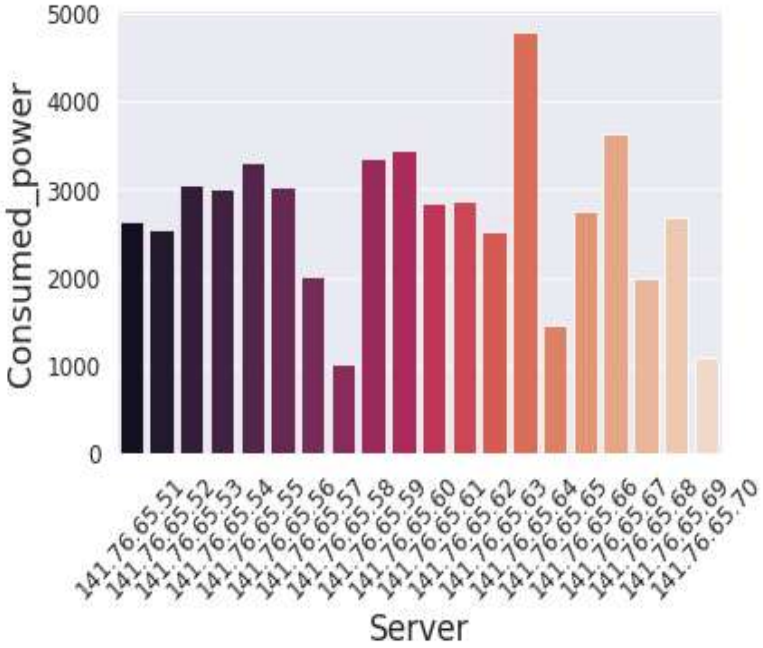


Рисунок Д4.2 Імітаційне моделювання: енергоспоживання серверів при використанні Round Robin

Додаток 5. Список публікацій за темою дисертації та відомості про апробацію результатів дисертації

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Globa, Larysa, Gvozdetska (*Prokopets*), Nataliia & Novogrudska, Rina. (2021). Ontological model for data processing organization in information and communication networks. *System research and information technologies*. pp. 47-60. ISSN: 1681–6048. DOI: https://doi.org/10.20535/SRIT.2308-8893.2021.1.04.

2. Прокопець, Н. А., Глоба, Л. С. (2022). Математичне моделювання процесу обслуговування навантаження в інформаційно-комунікаційній мережі, *Проблеми телекомунікацій*, (1 (30)), сс. 18-31. ISSN: 2220-6922. DOI: https://doi.org/10.30837/pt.2022.1.02.

3. Prokopets, N., Globa, L. (2022). Comprehensive method of energy-efficient workload processing in the information and communication network. *Information and Telecommunication Sciences*, (2), pp. 34-45. ISSN: 2312-4121. DOI: https://doi.org/10.20535/2411-2976.22022.34-45.

4. Глоба, Л. С., Гвоздецька (*Прокопець*), Н. А., Прокопець, В. А., & Степурін, О. В. (2017). Енергоефективний підхід до розподілу задач у серверному кластері. *Вісник Харківського національного університету імені В.Н. Каразіна*. Серія: Математичне моделювання. Інформаційні технології. Автоматизовані системи управління, (34), сс. 18-28. ISSN: 2524-2601.

5. Globa, L., Gvozdetska (*Prokopets*), N., Prokopets, V., Stryzhak, O. (2019). QoS and Energy Efficiency Improving in Virtualized Mobile Network EPC Based on Load Balancing. In: Pejaś, J., El Fray, I., Hyla, T., Kacprzyk, J. (eds) Advances in Soft and Hard Computing. ACS 2018. *Advances in Intelligent Systems and Computing*, vol 889. Springer, Cham. (pp. 140-149). ISSN: 2194-5357. DOI: https://doi.org/10.1007/978-3-030-03314-9_13

Наукові праці, які засвідчують апробацію матеріалів дисертації

6. Gvozdetska (*Prokopets*), N., Globa, L., & Prokopets, V. (2019). Energy-efficient Backfill-based scheduling approach for Slurm resource manager. In 2019 IEEE 15th

International Conference on the Experience of Designing and Application of CAD Systems (CADSM) (pp. 1-5). IEEE. ISSN: 2572-7591, DOI: https://doi.org/10.1109/CADSM.2019.8779312.

7. Globa, L., Gvozdetska (*Prokopets*), N. (2021). Experimental analysis of PCPB-2: Comprehensive energy-efficient approach to distributed workload processing in communication networks. In 2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-3). IEEE. DOI: https://doi.org/10.1109/BlackSeaCom52164.2021.9527759.

8. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Comprehensive energy efficient approach to workload processing in distributed computing environment. In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-6), DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235010.

9. Globa, L., Gvozdetska (*Prokopets*), N. (2020). Energy Efficient Workload Processing Technology for Distributed Computing Environment. In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-3). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom48709.2020.9235001.

10. Globa, L., Gvozdetska (*Prokopets*), N. (2019). Energy efficient workload processing in distributed computing environment modeling. In 2019 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 1-6). IEEE., DOI: https://doi.org/10.1109/UkrMiCo47782.2019.9165448.

11. Globa, L. S., Prokopets, V. A., & Gvozdetska (*Prokopets*), N. A. (2018). Prognostic-reactive NFV resource allocation method for implementation in virtualized mobile network EPC of Ukraine. In 2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 1-5). IEEE, DOI: https://doi.org/10.1109/BlackSeaCom.2018.8433602.

12. Schill, A., Globa, L., Stepurin, O., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2017). Power Consumption and Performance Balance (PCPB) scheduling algorithm for computer cluster. In 2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 1-8). IEEE.,

DOI: https://doi.org/10.1109/UkrMiCo.2017.8095365.

13. Gvozdetska (*Prokopets*), N., Stepurin, O., & Globa, L. (2017). Experimental analysis of PCPB scheduling algorithm. In 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM) (pp. 244-247). IEEE., DOI: https://doi.org/10.1109/CADSM.2017.7916126.

14. Globa, L., Gvozdetska (*Prokopets*), N., & Prokopets, V. (2021). Providing Energy-efficient and High-performance Infrastructure for Smart Network. In 2021 IEEE International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) (pp. 133-136). IEEE., DOI: https://doi.org/10.1109/UkrMiCo52950.2021.9716620.

Наукові праці, які додатково відображають наукові результати дисертації

15. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Оптимізація обчислень у дата-центрі за критерієм енергоефективності. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 127-150.

16. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба, В.А. Прокопець, & О.В. Степурін. (2019). Динамічний розподіл ресурсів у віртуалізованій мережі мобільного оператора України з підвищенням енергоефективності обробки даних. *Наукоємні технології оптимізації та керування в інфокомунікаційних мережах : розділ монографії*. Під загальною редакцією В. М. Безрука, Л. С. Глоби, О. Є. Стрижака. Київ : Інститут обдарованої дитини НАПН України. сс. 151-176.

17. Л.С. Глоба, Н.А. Гвоздецька (*Прокопець*), В.А. Прокопець, & О.В. Степурін (2017). Підхід до енергоефективного планування задач у серверному кластері для оптимізації обробки трафіка передачі даних. *Наукоемкие технологии в инфокоммуникациях: обработка информации, кибербезопасность, информационная борьба : розділ монографії*. Під загальною редакцією В. М. Безрука, В. В. Баранника. Х. : Издательство «Лидер», 2017. сс. 34-51.

18. L.Globa, O. Stryzhak, N. Gvozdetska (*Prokopets*), & V. Prokopets. (2019).

Intelligent Workload Scheduling in Distributed Computing Environment for Balance Between Energy Efficiency and Performance. *Scheduling Problems - New Applications and Trends*. London, 2019. (IntechOpen). 154 c. DOI: http://dx.doi.org/10.5772/intechopen.86874

19. Н.А. Гвоздецька (*Прокопець*), Л.С. Глоба. (2017). Підхід до оптимізації обробки даних з урахуванням параметрів енергоефективності та продуктивності. *Одинадцята міжнародна науково-технічна конференція "Проблеми телекомунікацій"*. 2017. сс. 464–466.

Відомості про апробацію результатів дисертації

Основні положення та результати дисертаційної роботи були представлені й одержали схвалення на:

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Батумі, Грузія, 2018 р.);

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Одеса, Україна, 2020 р.);

– Міжнародній науково-технічній конференції «Black Sea Conference on Communications and Networking (BlackSeaCom)» (м. Бухарест, Румунія, 2021 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Одеса, Україна, 2017 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Одеса, Україна, 2019 р.);

– Міжнародній науково-технічній конференції з Інформаційно-комунікаційних технологій та радіоелектроніки UkrMiCo (м. Київ, Україна, 2021 р.);

– Міжнародній науково-технічній конференції «The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)» (Поляна, Свалява, Україна, 2017 р.);

– Міжнародній науково-технічній конференції «The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)» (Поляна, Свалява, Україна, 2019 р.);

– Міжнародній науково-технічній конференції «International Conference on Advanced Computer Systems» (м. Мендзиздроє, Польща, 2018 р.);

– Міжнародній науково-технічній конференції «Проблеми телекомунікацій» (м. Київ, Україна, 2017 р).