НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова
праця на правах рукопису

Петровська Інна Юріївна

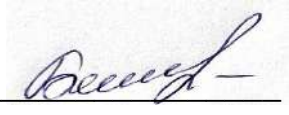
УДК 004.75:004.451.22 (043.3)

ДИСЕРТАЦІЯ
МЕТОДИ РОЗПОДІЛУ РЕСУРСІВ В КОМП'ЮТЕРНИХ СИСТЕМАХ ПРИ НАДАННІ ХМАРНИХ ІНФРАСТРУКТУРНИХ ПОСЛУГ

Спеціальність 123 – Комп'ютерна інженерія
Галузь знань 12 – Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

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

 І.Ю.Петровська

Науковий керівник:
Кучук Георгій Анатолійович,
доктор технічних наук, професор

Харків – 2023

АНОТАЦІЯ

Петровська І.Ю. Методи розподілу ресурсів в комп'ютерних системах при наданні хмарних інфраструктурних послуг. Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового доктора філософії (PhD) за спеціальністю 123 – Комп'ютерна інженерія. – Національний технічний інститут «Харківський політехнічний інститут», Україна, Харків, 2023.

Дисертаційна робота присвячена вирішенню актуальної науково-технічної задачі щодо підвищення ефективності використання хмарних обчислювальних ресурсів при застосуванні технології, що базується на моделі «Інфраструктура у якості сервісу», шляхом розробки методів розподілу ресурсів у хмарному середовищі..

Об'єкт дослідження – процес розподілу ресурсів у хмарному середовищі при використанні технології, що базується на моделі «Інфраструктура у якості сервісу».

Предмет дослідження – методи розподілу ресурсів у комп'ютерних системах, інфраструктура яких міститься у хмарному середовищі.

Метою дисертаційної роботи є підвищення ефективності використання хмарних обчислювальних ресурсів при використанні технології, що базується на моделі «Інфраструктура у якості сервісу» шляхом розробки методів розподілу ресурсів між користувачами хмари.

У вступі обґрунтовано актуальність розподілу ресурсів у хмарному середовищі, представлено зв'язок роботи з науковими програмами, планами і темами, наведено наукову новизну, представлено практичне значення отриманих результатів, надано інформацію щодо особистого внеску здобувача, представлено перелік публікацій за темою дисертації.

У першому розділі проведений аналіз методів розподілу ресурсів у хмарних середовищах. Зокрема, розглянуті Особливості хмарних обчислювальних систем та розподілу ресурсів в них. Проаналізовані існуючі статичні і динамічні методи розподілу ресурсів у віртуальних середовищах.

Проведений порівняльний аналіз підходів до розподілу ресурсів у хмарному середовищі з різними моделями обслуговування. На основі проведеного аналізу сформульовані задачі дослідження.

У другому розділі основна увага приділена завданню базового виділення ресурсів хмарного середовища з моделлю обслуговування «Інфраструктура у якості сервісу». Для його вирішення проведена декомпозиція хмарного середовища, обґрунтований вибір методу для базового виділення хмарних ресурсів та запропонований метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель IaaS. Також наведений приклад базового завантаження віртуального хоста за допомогою запропонованого метода.

У третьому розділі запропонований розгляд етапів методу адаптивного розподілу хмарних ресурсів при використанні моделі обслуговування «Інфраструктура у якості сервісу». Зокрема, проведене узагальнення вимог до методу розподілу хмарних ресурсів при використанні моделі обслуговування «Інфраструктура у якості сервісу» та запропонована структура методу адаптивного розподілу хмарних ресурсів. В межах даної структури розроблені метод адаптивного прогнозування на основі тесту на послідовність серій, метод превентивного формування черг запитів на віртуальні машини хмарного середовища та модель багатоцільового розподілу ресурсів, які дали можливість реалізувати адаптивний розподіл хмарних ресурсів у випадку використання моделі обслуговування «Інфраструктура у якості сервісу».

У четвертому розділі проведені дослідження запропонованих методів розподілу ресурсів в комп'ютерних системах при наданні хмарних інфраструктурних послуг. Зокрема, досліджені на імітаційній моделі методи прогнозування запитів на ресурси на основі тесту на послідовність серій та адаптивного розподілу хмарних ресурсів. Також наведені практичні рекомендації по використанню адаптивного методу розподілу хмарних ресурсів.

У висновках наведено основні результати наукової роботи щодо вирішення поставлених наукових задач дослідження.

За результатами дослідження отримано такі наукові результати:

1) отримав подальший розвиток метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу», шляхом попередньої декомпозиції множини доступних ресурсів на зони за допомогою введення нерівномірних шкал та використання методу аналізу ієрархій, що дозволяє підвищити рівень балансування завантаження хмарних ресурсів;

2) вперше розроблено метод превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», який враховує результати аналізу попередніх даних та базується на моделі багатоцільового розподілу хмарних ресурсів, що дозволяє завчасно провести прогнозування завантаженості фізичних пристроїв хмарного середовища та запобігти втратам обчислювального ресурсу;

3) удосконалено метод адаптивного розподілу ресурсів хмарного середовища, який відрізняється від відомих використанням тестування на послідовність серій, математичного апарату удосконаленого генетичного алгоритму NSGA-II та результатами прогнозу запитів на віртуальні машини, що дозволяє підвищити ефективність використання хмарних обчислювальних ресурсів за рахунок реалізації балансу між ресурсами центрального процесора та оперативної пам'яті та зменшення затримки в обслуговуванні хмарних ресурсів.

Практичне значення отриманих результатів полягає в тому, що розроблені у роботі методи є науково-практичною основою для подальшого удосконалення хмарного середовища, при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу». Представлені на їх основі інженерні методи та алгоритми дають змогу:

– провести короткочасне прогнозування можливих запитів на обчислювальні ресурси хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»;

– сформувати можливі черги запитів на найбільш витратний за часом формування хмарний ресурс – віртуальні машини, з відхиленням від реальних запитів не більше 15%;

– підвищити рівень балансування завантаження обчислювальних ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», за показником середнього квадратичного відхилення до 8%;

– зменшити затримку в обслуговуванні обчислювальних ресурсів до 5%.

За результатами дослідження підтверджено практичну та теоретичну цінність розроблених методів, надано практичні рекомендації, щодо застосування розроблених методів та розглянуто перспективи їх подальшого розвитку.

Ключові слова: комп’ютерна система, хмарні обчислення, розподіл ресурсів, багатопроцесорна система, система підтримки прийняття рішень, інфокомунікаційна мережа, нейронна мережа, метод аналізу ієрархій, балансування, буферна пам’ять, тестування, сумісність, транзакція, обчислювальний вузол.

Список публікацій здобувача

Наукові праці, в яких опубліковано основні наукові результати:

1. Петровська І. Ю., Кучук Н. Г., Панченко В. І., Філоненко А. М. Рівномірний розподіл ресурсів комп'ютерних систем, що мають гіперконвергентну інфраструктуру. *Системи управління, навігації та зв’язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2019, Вип. 2 (54). С. 119–122. DOI: 10.26906/SUNZ.2019.2.119.

2. Петровська І. Ю., Коломійцев О. В., Алнаері Фрхат Алі. Метод розрахунку розміру буферної пам’яті самовідновлювального сегмента

телекомунікаційної мережі. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2021, вип. 2 (64). С. 144–147. DOI: 10.26906/SUNZ.2021.2.144.

3. Petrovska Inna, Kuchuk Heorhii. Static allocation method in a clod environment with a service model IAAS. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2022, Т. 6, № 3. С. 99–105. DOI: 10.20998/2522-9052.2022.3.13.

4. Петровська І. Ю., Кучук Г. А. Розподіл обчислювальних ресурсів у хмарних системах. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2022, вип. 2 (68). С. 75–78.

DOI: 10.26906/SUNZ.2022.2.75.

5. Petrovska Inna, Kuchuk Heorhii. Adaptive resource allocation method for data processing and security in cloud environment. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2023, Т. 7, № 3. С. 67–73.

DOI: 10.20998/2522-9052.2023.3.10.

Опубліковані праці апробаційного характеру:

6. Петровська І. Ю. Методи розподілу ресурсів в хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей VII Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2019. С. 75.

7. Петровська І. Ю., Заполовський М. Й., Шемякін Є. Ю. Система автоматизованного тестування серверної частини мобільного додатку. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали X Міжнародної науково-технічної конференції . Баку – Харків – Жиліна, 2020, Т. 2. С.15.

8. Петровська І. Ю., Заполовський М. Й., Мітяєв А. С. Розроблення та дослідження мобільного додатку на основі фреймворку REACT NATIVE. *Проблеми інформатизації* : Тези доповідей IIX Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2020, Т. 2. С. 23.

9. Петровська І. Ю., Кучук Г. А., Кучук Н.Г. Підходи до розподілу ресурсів у хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей IX Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2021, Т. 2. С. 48.

10. Петровська І. Ю., Кучук Г. А. Особливості розподілу ресурсів в хмарних обчислювальних середовищах. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали XII Міжнародної науково-технічної конференції. Баку – Харків – Жиліна, 2022. Т. 1. С. 26.

11. Petrovska Inna, Yefymenko Serhii, Hryhorenko Ihor, Khoroshilo Iurii, Hryhorenko Svitlana. Evaluation of informativeness of indicators in colorimetric control using discriminative analysis. *32 International Scientific Symposium on MMA*. Созопіль, Болгарія, 2022. DOI: 10.1109/MMA55579.2022.9992712.

12. Petrovska Inna, Kuchuk Heorhii, Mozhaiev Mykhailo. Features of the distribution of computing resources in cloud systems. *2022 IEEE 3rd KhPI Week on Advanced Technology Conference*. Харків, 2022.

DOI: 10.1109/KhPIWeek57572.2022.9916459.

13**.** Kuchuk H., Petrovska I. Modeling data processing programs in the self-healing network. *15th International symposium of Croatian metallurgical society SHMD*. Загреб, Хорватія, 2022. С. 575.

14. Петровська І. Ю., Кучук Г. А. Порівняння хмарних та туманних обчислень для Інтернету Речей. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали XIII Міжнародної науково-технічної конференції. Баку – Харків – Жиліна, 2023, Т. 2. С.53.

15. Rezanov B., Semenova A., Petrovska I., Fesenko T. Model for Providing the Second Factor of Authentication Into Authentication Services with Centralized Account Databases. Proceedings of the 5th International Scientific and Technical Conference "Computer and Information Systems and Technologies". Харків, ХНУРЕ, 2021. С. 46-47.

URL: http://csitic.nure.ua/article/view/232201.

ABSTRACT

Inna Petrovska. Methods of resource allocation in computer systems when providing cloud infrastructure services. Qualifying scientific work on manuscript rights.

Dissertation for obtaining a scientific doctor of philosophy (PhD) in specialty 123 - Computer engineering. - National Technical Institute "Kharkiv Polytechnic Institute", Ukraine, Kharkiv, 2023.

The dissertation is devoted to solving the current scientific problem of developing methods of resource allocation in a cloud computing environment to increase the efficiency of their use and minimize costs when using technology based on the "Infrastructure as a service" model.

The object of research is the process of resource allocation in a cloud computing environment using technology based on the "Infrastructure as a service" model.

The subject of research is methods of resource allocation in computer systems, the infrastructure of which is contained in a cloud computing environment.

The aim of the dissertation work is to increase the efficiency of cloud computing resources when using technology based on the "Infrastructure as a service" model by developing methods of resource distribution among cloud users.

The introduction substantiates the relevance of the distribution of resources in the cloud computing environment, presents the connection of the work with scientific programs, plans and topics, provides scientific novelty, presents the practical significance of the obtained results, provides information on the personal contribution of the recipient, presents a list of publications on the topic of the dissertation.

In the first chapter, an analysis of resource allocation methods in cloud computing environments is carried out. In particular, the Peculiarities of cloud computing systems and the distribution of resources in them are considered. Existing static and dynamic methods of resource allocation in virtual environments are analyzed. A comparative analysis of approaches to resource allocation in a cloud

computing environment with different service models was carried out. Based on the analysis, the research objectives were formulated.

In the second chapter, the main attention is paid to the task of basic resource allocation of the cloud environment with the service model "Infrastructure as a service". To solve it, the decomposition of the cloud computing environment, the justified choice of the method for the basic allocation of cloud resources, and the proposed method of the basic allocation of resources of the cloud environment to the user, oriented on the IaaS model, were made. An example of a basic virtual host load using the proposed method is also provided.

In the third section, consideration of the stages of the method of adaptive distribution of cloud resources when using the service model "Infrastructure as a service" is proposed. In particular, a generalization of the requirements for the method of cloud resource distribution when using the service model "Infrastructure as a service" was carried out and the structure of the method of adaptive distribution of cloud resources was proposed. Within the framework of this structure, the method of adaptive forecasting based on the series sequence test, the method of preventive formation of queues of requests for virtual machines of the cloud computing environment and the model of multi-purpose distribution of resources were developed, which made it possible to implement the adaptive distribution of cloud resources in the case of using the service model "Infrastructure as a service" .

In the fourth chapter, studies of the proposed methods of resource allocation in computer systems for the provision of cloud infrastructure services are carried out. In particular, the methods of forecasting requests for resources based on the series sequence test and adaptive distribution of cloud resources were studied on a simulation model. Practical recommendations for using the adaptive method of cloud resource allocation are also given.

In the conclusions, the main results of the scientific work regarding the solution of the set scientific problems of the research are presented.

According to the results of the research, the following scientific results were obtained:

1) the method of basic resource allocation of the cloud computing environment to the user oriented to the "Infrastructure as a service" model was further developed by preliminary decomposition of the set of available resources into zones using the introduction of uneven scales and the use of the method of analysis of hierarchies, which allows to increase the level of load balancing of cloud resources ;

2) for the first time, a method of preventive formation of queues of requests for virtual machines of the cloud computing environment was developed using technology oriented to the "Infrastructure as a service" model, which takes into account the results of the analysis of previous data and is based on a model of multi-purpose distribution of cloud resources, which allows for early forecasting of the load of physical cloud environment devices and prevent loss of computing resources;

3) the method of adaptive distribution of resources of the cloud computing environment has been improved, which differs from the known ones by the use of series sequence testing, the mathematical apparatus of the improved genetic algorithm NSGA-II and the results of the forecast of requests for virtual machines, which allows to increase the efficiency of the use of cloud computing resources due to the implementation of a balance between resources of the central processor and RAM and reducing the delay in servicing cloud resources.

The practical significance of the obtained results is that the methods developed in the work are a scientific and practical basis for the further improvement of the cloud computing environment, when using technology focused on the "Infrastructure as a service" model. The engineering methods and algorithms presented on their basis enable:

- carry out short-term forecasting of possible requests for computing resources of the cloud environment when using technology focused on the "Infrastructure as a service" model;

- form possible queues of requests for the most time-consuming cloud resource - virtual machines, with a deviation from real requests of no more than 15%;

- increase the level of load balancing of computing resources of the cloud environment when using technology focused on the "Infrastructure as a service" model, based on the average squared deviation of up to 8%;

- reduce the delay in servicing computing resources to 5%.

Based on the results of the study, the practical and theoretical value of the developed methods was confirmed, practical recommendations were given regarding the application of the developed methods, and the prospects for their further development were considered.

Keywords: computer system, cloud computing, resource allocation, multiprocessor system, decision support system, information communication network, neural network, hierarchical analysis method, balancing, buffer memory, testing, interoperability, transaction, computing node.

List of publications of the acquirer

Scientific works in which the main scientific results were published:

1. Petrovska I. Yu., Kuchuk N. H., Panchenko V. I., Filonenko A. M. Rivnomirnyi rozpodil resursiv kompiuternykh system, shcho maiut hiperkonverhentnu infrastrukturu. *Systemy upravlinnia, navihatsii ta zviazku*. Poltava: Natsionalnyi universytet «Poltavska politekhnika imeni Yuriia Kondratiuka», 2019, Vyp. 2 (54). C. 119–122.

DOI: 10.26906/SUNZ.2019.2.119.

2. Petrovska I. Yu., Kolomiitsev O. V., Alnaeri Frkhat Ali. Metod rozrakhunku rozmiru bufernoi pamiati samovidnovliuvalnoho sehmenta telekomunikatsiinoi merezhi. Systemy upravlinnia, navihatsii ta zviazku. Poltava: Natsionalnyi universytet «Poltavska politekhnika imeni Yuriia Kondratiuka», 2021, Vyp.2 (64). C. 144–147.

DOI: 10.26906/SUNZ.2021.2.144.

3. Petrovska Inna, Kuchuk Heorhii. Static allocation method in a clod environment with a service model IAAS. *Modern information systems. Kharkiv:* NTU "KhPI", 2022, T. 6, № 3. C. 99–105. DOI: 10.20998/2522-9052.2022.3.13.

4. Petrovska I. Yu., Kuchuk H. A. Rozpodil obchysliuvalnykh resursiv u khmarnykh systemakh. *Systemy upravlinnia, navihatsii ta zviazku*. Poltava: Natsionalnyi universytet «Poltavska politekhnika imeni Yuriia Kondratiuka», 2022, Vyp.2 (68). C. 75–78.

DOI: 10.26906/SUNZ.2022.2.75.

5. Petrovska Inna, Kuchuk Heorhii. Adaptive resource allocation method for data processing and security in cloud environment. *Modern information systems. Kharkiv:* NTU "KhPI", 2023, T. 7, № 3. C. 67–73.

DOI: 10.20998/2522-9052.2023.3.10.

Опубліковані праці апробаційного характеру:

6. Petrovska I. Yu. Metody rozpodilu resursiv v khmarnykh obchysliuvalnykh seredovyshchakh. *Problemy informatyzatsii* : Tezy dopovidei VII Mizhnarodnoi naukovo-tekhnichnoi konferentsii. Cherkasy – Baku – Belsko-Biala – Kharkiv, 2019. C. 75.

7. Petrovska I. Yu., Zapolovskyi M. Y., Shemiakin Ye. Yu. Systema avtomatyzovannoho testuvannia servernoi chastyny mobilnoho dodatku. *Suchasni napriamy rozvytku informatsiino-komunikatsiinykh tekhnolohii ta zasobiv upravlinnia*: Materialy X Mizhnarodnoi naukovo-tekhnichnoi konferentsii . Baku – Kharkiv – Zhylina, 2020, T. 2. C.15.

8. Petrovska I. Yu., Zapolovskyi M. Y., Mitiaiev A. S. Rozroblennia ta doslidzhennia mobilnoho dodatku na osnovi freimvorku REACT NATIVE. *Problemy informatyzatsii:* Tezy dopovidei IIX Mizhnarodnoi naukovo-tekhnichnoi konferentsii. Cherkasy – Baku – Belsko-Biala – Kharkiv, 2020, T. 2. C. 23.

9. Petrovska I. Yu., Kuchuk H. A., Kuchuk N.H. Pidkhody do rozpodilu resursiv u khmarnykh obchysliuvalnykh seredovyshchakh. *Problemy informatyzatsii*: Tezy dopovidei IX Mizhnarodnoi naukovo-tekhnichnoi konferentsii. Cherkasy – Baku – Belsko-Biala – Kharkiv, 2021, T. 2. C. 48.

10. Petrovska I. Yu., Kuchuk H. A. Osoblyvosti rozpodilu resursiv v khmarnykh obchysliuvalnykh seredovyshchakh. *Suchasni napriamy rozvytku informatsiino-*

komunikatsiinykh tekhnolohii ta zasobiv upravlinnia: Materialy XII Mizhnarodnoi naukovo-tekhnichnoi konferentsii. Baku – Kharkiv – Zhylina, 2022, T. 1. C.26.

11. Petrovska Inna, Yefymenko Serhii, Hryhorenko Ihor, Khoroshilo Iurii, Hryhorenko Svitlana. Evaluation of informativeness of indicators in colorimetric control using discriminative analysis. *32 International Scientific Symposium on MMA.* Sozopil, Bolhariia, 2022.

DOI: 10.1109/MMA55579.2022.9992712.

12. Petrovska Inna, Kuchuk Heorhii, Mozhaiev Mykhailo. Features of the distribution of computing resources in cloud systems. *2022 IEEE 3rd KhPI Week on Advanced Technology Conference*. Kharkiv, 2022.

DOI: 10.1109/KhPIWeek57572.2022.9916459.

13**.** Kuchuk H., Petrovska I. Modeling data processing programs in the self-healing network. *15th International symposium of Croatian metallurgical society SHMD*. Zahreb, Khorvatiia, 2022. C. 575.

14. Petrovska I. Yu., Kuchuk H. A. Porivniannia khmarnykh ta tumannykh obchyslen dlia Internetu Rechei. Suchasni napriamy rozvytku informatsiino-komunikatsiinykh tekhnolohii ta zasobiv upravlinnia : Materialy XIII Mizhnarodnoi naukovo-tekhnichnoi konferentsii. Baku – Kharkiv – Zhylina, 2023. T. 2. C.53.

15. Rezanov B., Semenova A., Petrovska I., Fesenko T. Model for Providing the Second Factor of Authentication Into Authentication Services with Centralized Account Databases. Proceedings of the 5th International Scientific and Technical Conference "Computer and Information Systems and Technologies". Kharkiv, ХНУРЕ, 2021. С. 46-47. URL: http://csitic.nure.ua/article/view/232201.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 4

ВСТУП 6

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗПОДІЛУ РЕСУРСІВ У ХМАРНИХ СЕРЕДОВИЩАХ 13

1.1. Особливості хмарних обчислювальних систем 13

1.2. Особливості розподілу ресурсів в хмарних обчислювальних системах 16

1.3. Методи розподілу ресурсів у віртуальних середовищах 22

1.3.1 Динамічні методи розподілу базового навантаження 23

1.3.2 Статичні методи розподілу базового навантаження 28

1.4. Порівняльний аналіз підходів до розподілу ресурсів у хмарному середовищі з різними моделями обслуговування 31

1.5. Обґрунтування завдання підвищення ефективності використання ресурсів хмарного середовища. 35

1.6. Висновки за розділом 1 37

РОЗДІЛ 2. РОЗРОБКА МЕТОДУ БАЗОВОГО ВИДІЛЕННЯ РЕСУРСІВ ХМАРНОГО СЕРЕДОВИЩА З МОДЕЛЛЮ ОБСЛУГОВУВАННЯ «ІНФРАСТРУКТУРА У ЯКОСТІ СЕРВІСУ» 38

2.1. Декомпозиція хмарного середовища 39

2.2. Обґрунтування вибору методу для базового виділення хмарних ресурсів 42

2.3. Метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель IaaS 47

2.4. Приклад базового завантаження віртуального хоста 52

2.5. Висновки за розділом 2 59

РОЗДІЛ 3. РОЗРОБКА МЕТОДУ АДАПТИВНОГО РОЗПОДІЛУ ХМАРНИХ РЕСУРСІВ ПРИ ВИКОРИСТАННІ МОДЕЛІ ОБСЛУГОВУВАННЯ «ІНФРАСТРУКТУРА У ЯКОСТІ СЕРВІСУ» 61

3.1. Узагальнення вимог до методу розподілу хмарних ресурсів при використанні моделі обслуговування «Інфраструктура у якості сервісу» 62

3.2. Структура методу адаптивного розподілу хмарних ресурсів 66

3.3. Метод адаптивного прогнозування на основі тесту на послідовність серій 69

3.4. Метод превентивного формування черг запитів на віртуальні машини хмарного середовища 77

3.5. Модель багатоцільового розподілу ресурсів 81

3.6. Висновки за розділом 3 92

РОЗДІЛ 4. ДОСЛІДЖЕННЯ ЗАПРОПОНОВАНИХ МЕТОДІВ РОЗПОДІЛУ РЕСУРСІВ В КОМП'ЮТЕРНИХ СИСТЕМАХ ПРИ НАДАННІ ХМАРНИХ ІНФРАСТРУКТУРНИХ ПОСЛУГ 95

4.1. Дослідження методу прогнозування запитів на ресурси на основі тесту на послідовність серій 95

4.2. Дослідження методу адаптивного розподілу хмарних ресурсів 105

4.3. Практичні рекомендації по використанню адаптивного методу розподілу хмарних ресурсів 111

4.4. Висновки за розділом 4 114

ВИСНОВКИ 116

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 120

ДОДАТОК А СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ 135

ДОДАТОК Б АКТИ ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ НАУКОВИХ ДОСЛІДЖЕНЬ ДИСЕРТАЦІЙНОЇ РОБОТИ 138

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CaaS − Calculations as a Service (розрахункові обчислення як сервіс);

CPU − Central Processing Unit;

DPM − Distributed Power;

DRS − Dirstibuted Resource Scheduling;

HZ − зона високого навантаження (High load zone);

IaaS − Infrastructure as a Service (інфраструктура як послуга);

IQR – міжквартильний розмах;

LZ − зона низького навантаження (Low load zone);

MZ − зона середнього навантаження (Medium load zone);

PaaS − Platform as a Service (платформа як послуга);

RAM − Random Access Memory;

SaaS − Software as a Service (програмне забезпечення як послуга);

КС – комп’ютерна система;

ПЗ – програмне забезпечення;

ХКС – хмарні комп'ютерні системи;

ХОС – хмарна обчислювальна система;

ХС – хмарне середовище;

ЦП − центральний процесор;

N_{CPU} – число CPU;

N_{Nucl} – кількість ядер в кожному CPU;

V_{CPU} – тактова частота CPU;

V_{net} – швидкість роботи мережі;

ΔT_{fail} – середній час між збоями;

$Cond_{hw}$ – ступінь фізичного стану Hardware;

$Serv_{pers}$ – рівень компетенцій персоналу для обслуговування обладнання;

VM – середня швидкість доступу до зовнішньої пам'яті;

EM – обсяг доступної зовнішньої пам'яті;

V_{RAM} – вільна область RAM;

Z_{CPU} – завантаження CPU;

Z_{net} – завантаження мережі;

N_{vh} – кількість екземплярів віртуальних хостів;

c_{rent} – відносна вартість місячної оренди.

ВСТУП

Актуальність теми. Розвиток інформаційного суспільства привносить якісні корективи до уявлень користувачів про те, як повинні надаватися інформаційні сервіси: на перший план виходять не технічні характеристики послуг, що надаються, а якісні показники: своєчасне задоволення потреб бізнесу, простота використання, швидкість виконання типових операцій та ін.

На сьогодні, найбільш затребуваними для користувачів є хмарні технології, причому взаємодія користувача та постачальника ресурсів відбувається за однією із моделей, за якими отримуються або розрахункові послуги, або надається необхідне програмне забезпечення, або якась обчислювальна платформа. Але у останні кілька років користувачі, які мали великі обчислювальні потужності, як апаратні, так і програмні, стали зацікавленими у отриманні власної хмарної віртуальної інфраструктури. У зв'язку з цим, для такої категорії користувачів визначальним підходом при наданні обчислювальних потужностей стає модель хмарної обчислювальної системи (ХОС) "Інфраструктура у якості сервісу" (IaaS), яка дозволяє звести до мінімуму взаємодію постачальника і споживача обчислювальних ресурсів (під обчислювальними ресурсами розуміються можливості, що забезпечуються компонентами обчислювальної системи, які витрачаються (займані) в процесі її роботи) з технічних питань. При цьому скорочується число інцидентів і час їх обробки, що надає бізнесу можливість адаптувати роботу під власні потреби, а також скорочуються капітальні і операційні затрати.

Усередині такого хмарного середовища (багатомашинного обчислювального комплексу, що функціонує на основі моделі IaaS) можуть розміщуватися найрізноманітніші віртуальні системи – починаючи від малонавантаженого вебсайту і закінчуючи високонавантаженими базами даних, ERP-системами та ін. Очевидно, що ці системи пред'являють абсолютно різні вимоги до ресурсів, отже, розміщувати їх на одних і тих же хостах, сховищах та

ділянках мережі недоцільно – це призведе до нераціонального використання ресурсів, зниження продуктивності та додаткових витрат.

Хмарне середовище містить у собі великий набір обчислювальних ресурсів: сервери, системи зберігання даних, мережні пристрої. При цьому вони можуть бути як однотипними, так і різними за продуктивністю, набором інструкцій, співвідношенням числа процесорних ядер до обсягу оперативної пам'яті тощо. Це ускладнює проблему оптимального використання наявних ресурсів, призводячи до значного подорожчання, ускладнення управління або навіть повної неможливості використання хмарних обчислювальних середовищ, функціонуючих за технологією на базі моделі IaaS, для вирішення різнорідних завдань на одному наборі обладнання.

Таким чином**, актуальною науково-технічною задачею** є підвищення ефективності використання хмарних обчислювальних ресурсів при застосуванні технології, що базується на моделі «Інфраструктура у якості сервісу», шляхом розробки методів розподілу ресурсів у хмарному середовищі. У цьому мають враховуватися як реальні потреби застосунків користувачів хмарного середовища, так і показники використання наявних ресурсів з урахуванням їхньої специфіки.

Зв’язок роботи з науковими програмами, планами, темами. Дослідження виконані на кафедрі «Комп’ютерна інженерія та програмування» НТУ «Харківський політехнічний інститут» в рамках двох науково-дослідних тем: «Моделі і методи обробки та захисту інформації в комп'ютерних системах» (ДР №0122U200526, ТОВ «Передові цифрові рішення», м. Харків), «Моделі і методи обробки даних і розподілу мережних ресурсів в комп'ютерних системах» (ДР №0122U200527, компанія «LineUp», м. Харків), в яких здобувач брав участь у якості виконавця окремих розділів.

Об’єкт дослідження – процес розподілу ресурсів у хмарному середовищі при використанні технології, що базується на моделі «Інфраструктура у якості сервісу».

Предмет дослідження – методи розподілу ресурсів у комп'ютерних системах, інфраструктура яких міститься у хмарному середовищі.

Мета дослідження – підвищення ефективності використання хмарних обчислювальних ресурсів при використанні технології, що базується на моделі обслуговування «Інфраструктура у якості сервісу» шляхом розробки методів розподілу ресурсів між користувачами хмарного середовища.

Відповідно до поставленої мети визначені наступні задачи:

1) проведення аналізу існуючих методів розподілу ресурсів у хмарних середовищах з метою виявлення проблем, що виникають при використанні моделі обслуговування «Інфраструктура у якості сервісу»;

2) розроблення методу базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу»;

3) розроблення методу превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»;

4) розроблення методу адаптивного розподілу ресурсів хмарного середовища;

5) проведення порівняльної оцінки розроблених та існуючих методів розподілу ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»;

6) дослідження та впровадження розроблених методів розподілу ресурсів хмарного середовища.

Методи дослідження. При розв’язанні наукової задачі використано широкий спектр методів. Так, при розробці базового виділення та розподілу ресурсів хмарного середовища використовувалися: теоретико-множинний підхід; метод аналізу ієрархій, методи теорії масового обслуговування, дослідження операцій та еволюційні методи. Оцінка експериментальних даних, отриманих у ході роботи, проводилася на основі методів теорії ймовірності та математичної статистики.

Вибір методів досліджень забезпечив достовірність отриманих результатів і висновків, що підтверджується збіжністю результатів експериментальних досліджень, отриманих при програмній реалізації алгоритмів розподілу ресурсів хмарного середовища з теоретичними і практичними результатами.

Наукова новизна отриманих результатів обумовлена розробленими методами базового виділення та розподілу ресурсів хмарного середовища, в межах яких отримані такі нові наукові результати:

1) отримав подальший розвиток метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу», шляхом попередньої декомпозиції множини доступних ресурсів на зони за допомогою введення нерівномірних шкал та використання методу аналізу ієрархій, що дозволяє підвищити рівень балансування завантаження хмарних ресурсів;

2) вперше розроблено метод превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», який враховує результати аналізу попередніх даних та базується на моделі багатоцільового розподілу хмарних ресурсів, що дозволяє завчасно провести прогнозування завантаженості фізичних пристроїв хмарного середовища та запобігти втратам обчислювального ресурсу;

3) удосконалено метод адаптивного розподілу ресурсів хмарного середовища, який відрізняється від відомих використанням тестування на послідовність серій, математичного апарату удосконаленого генетичного алгоритму NSGA-II та результатами прогнозу запитів на віртуальні машини, що дозволяє підвищити ефективність використання хмарних обчислювальних ресурсів за рахунок реалізації балансу між ресурсами центрального процесора та оперативної пам'яті та зменшення затримки в обслуговуванні хмарних ресурсів.

Практичне значення отриманих результатів полягає в тому, що розроблені у роботі методи є науково-практичною основою для подальшого удосконалення хмарного середовища, при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу». Представлені на їх основі інженерні методи та алгоритми дають змогу:

– провести короткочасне прогнозування можливих запитів на обчислювальні ресурси хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»;

– сформувати можливі черги запитів на найбільш витратний за часом формування хмарний ресурс – віртуальні машини, з відхиленням від реальних запитів не більше 15%;

– підвищити рівень балансування завантаження обчислювальних ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», за показником середнього квадратичного відхилення до 8%;

– зменшити затримку в обслуговуванні обчислювальних ресурсів до 5%.

Результати досліджень впроваджено (див. додаток Б):

– на підприємстві ТОВ «Лайнап Тех» (м. Харків) в ході дослідницьких робіт при проектуванні комплексної рекомендаційної системи (акт впровадження від 20.06.2023 р.);

– у навчальний процес кафедри “Комп’ютерна інженерія та програмування” Національного технічного університету “ХПІ” (акт впровадження від 05.10.2023 р.).

Особистий внесок здобувача полягає в розробленні нових методів і рішень, які забезпечують виконання поставлених в дисертаційній роботі завдань.

У роботах, опублікованих у співавторстві, здобувачем належать такі результати: метод рівномірного розподілу ресурсів у комп’ютерних системах [24]; метод розподілу буферної пам’яті [25]; метод статичного розподілу ресурсів у хмарних системах для моделі IaaS [26, 32, 97]; метод адаптивного

розподілу ресурсів у хмарних системах для моделі IaaS [98, 100]; порівняльний аналіз методів розподілу ресурсів у хмарних системах [27, 30, 31]; модель хмарного середовища для проведення тестування [28, 29, 99]; модель багатоцільового розподілу ресурсів [101].

Апробація результатів. Основні наукові положення й результати, отримані автором при виконанні дисертаційної роботи, доповідалися на:

VII Міжнародній науково-технічної конференції «Проблеми інформатизації». Черкаси – Баку – Бельсько-Бяла – Харків, 2019;

VIII Міжнародній науково-технічної конференції «Проблеми інформатизації». Черкаси – Баку – Бельсько-Бяла – Харків, 2020;

IX Міжнародній науково-технічної конференції «Проблеми інформатизації». Черкаси – Баку – Бельсько-Бяла – Харків, Т. 2. 2021;

X Міжнародній науково-технічної конференції «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління». Баку – Харків – Жиліна, 2020;

XII Міжнародній науково-технічної конференції. Баку – Харків – Жиліна, Т. 1. «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління» Баку – Харків – Жиліна, 2022 р.;

XIII Міжнародній науково-технічної конференції «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління». Баку – Харків – Жиліна, 2023;

XXXII Міжнародній науково-технічної конференції «International Scientific Symposium on Metrology and metrology assurance» Болгарія,Созополь, 2022 р. (SCOPUS);

III Міжнародній науково-технічної конференції «IEEE 3rd KhPI Week on Advanced Technology Conference» Україна, Харків, 2022 р. (SCOPUS);

XXII Міжнародній науково-технічної конференції «International symposium of Croatian metallurgical society SHMD» Хорватія, Загреб, 2022 р.

V Міжнародній науково-технічної конференції "Computer and Information Systems and Technologies". Україна, Харків : ХНУРЕ, 2021р.

Публікації. Результати наукових досліджень відображено в 15 друкованих працях, зокрема в 5 статтях у наукових фахових виданнях України, 8 публікаціях в матеріалах міжнародних наукових конференцій, 2 публікаціях в матеріалах міжнародних наукових конференцій внесені до міжнародної наукометричної бази SCOPUS.

Структура і обсяг дисертації. Дисертаційна робота складається з анотації двома мовами, вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Робота містить 114 сторінок основного тексту, серед них: 40 рисунків, 8 таблиць, 126 найменувань у списку використаних джерел, 5 сторінок додатків. Загальний обсяг роботи викладено на 139 сторінках.

РОЗДІЛ 1
АНАЛІЗ МЕТОДІВ РОЗПОДІЛУ РЕСУРСІВ У ХМАРНИХ СЕРЕДОВИЩАХ

1.1 Особливості хмарних обчислювальних систем

Розвиток інформаційної індустрії змінив уявлення користувачів (власників бізнесу) про надання інформаційних послуг. На перше місце виходять не тільки технічні характеристики сервісів, але й показники якості. Вони демонструють поточну задоволеність потреб бізнесу, легкість підбору, швидкість виконання стандартних операцій тощо. [52, 56]. У зв'язку з цим, хмарні обчислення набувають все більшої популярності (англ. cloud computing).

Хмарні обчислення надають можливість отримувати обчислювальний ресурс через Інтернет з такими обов'язковими характеристиками [126]:

- самообслуговування за запитом (англ. self service on demand) − споживач має можливість самостійно визначати та змінювати потреби в обчисленнях, а саме: швидкості як доступу, так і обробки даних, серверний час, обсяг даних, що зберігаються, при цьому немає необхідності взаємодіяти з постачальниками, що надають послуги;
- універсальний доступ через мережу − через мережу передачі даних користувачі, незалежно від наявного пристрою, отримають необхідні послуги;
- об'єднання ресурсів (англ. resource pooling) − ресурси для одночасного бслуговування великої кількості споживачів, постачальники послуг об'єднують у єдиний пул для подальшого динамічного перерозподілу потужностей, за умов постійної зміни запитів на потужності, між споживачами; при цьому користувачі контролюють тільки такі параметри, як швидкість доступу, обсяг даних тощо [62]; по факту, розподіл ресурсів, котрі отримують користувачі хмари, здійснюється постачальником, хоча у обмежених випадках, управляти якимось параметрами перерозподілу [72] користувачі все ж можуть, наприклад, вони можуть вказати конкретний центр для обробки своїх даних (англ. data center), орієнтуючись, наприклад, на географічну близькість;

- еластичність − об’єми послуги можна розширювати, або звужені будь-якої миті, без додаткових витрат [61];
- облік споживання − на конкретному рівні абстракції (пропускна спроможність, обсяг даних, що зберігаються, кількість транзакцій, кількість користувачів) постачальники послуг можуть автоматично обчислити об’єми спожитих ресурсів, і, базуючись на отриманих даних, вони мають змогу оцінити об’єми послуг, що надані користувачам [64].

Суть хмарних технологій є у тому, що з їхньою допомогою вдається надавати доступ до будь-яких конфігурацій обчислювальних ресурсів. Маються на увазі сервери, мережі, застосунки, сховища тощо. Все це можна легко і швидко використовувати або звільнити [68]. Управління є нескладним, тому при цьому не потрібно безпосередньо контактувати з провайдером хмарних послуг.

Визначають такі переваги хмарних технологій [1, 3, 20]:

Зниження витрат. Коли є доступ до хмарних структур, власне дороге обладнання більше не потрібно купувати, ні обслуговувати. Витрати на комунальні послуги знижуються, і в масштабному центрі обробки даних також немає потреби.

Дані перебувають у більшій безпеці. Збереження цінної інформації – одне з найважливіших завдань у будь-якому бізнесі, не має значення, наскільки він масштабний і у якій сфері він ведеться. Будь-які кіберзлочини, у тому числі і крадіжка цінної інформації, здатні підкосити діяльність компанії, завдати шкоди бренду, репутації. Широкий функціонал безпеки хмарних технологій забезпечують більш надійне зберігання та обробку даних.

Можливість масштабування. ІТ-потреби в різних компаній неоднакові. Стартапу необхідне одне, а великому підприємству, звісно, зовсім інше. Але у хмарі можна використовувати стільки ІТ-ресурсів, скільки потрібно, знижувати або збільшувати їх масштаби з урахуванням потреб.

Мобільний доступ. Цінна на сьогоднішній день опція, що дозволяє використовувати корпоративні дані у будь-який момент (через смартфон чи

інший гаджет), перебуваючи далеко від офісу, десь у дорозі тощо. Так кожен співробітник може навіть за дуже щільного робочого графіка постійно «тримати руку на пульсі».

Можливість аварійного відновлення. Зрозуміло, що ніхто не зацікавлений у втраті даних. Сервіси хмарних технологій якраз гарантують, що такого не станеться, за будь-яких непередбачених ситуацій дані дуже швидко відновлюються.

Ступінь контролю. Звичайно, будь-яка компанія зацікавлена в тому, щоб мати максимальний контроль над власною конфіденційною інформацією. Можна самостійно контролювати доступ до даних, що зберігаються у хмарі.

До недоліків хмарних обчислень відносяться [18, 20]:

Паузи у роботі. Зрозуміло, власники хмарних сервісів намагаються залучати якомога більше клієнтів. В результаті через перевантаженість в системі нерідкі технічні збої, що загрожує простоями у бізнесі [71].

Рівень безпеки. Так, тут, як правило, задіяні найсучасніші стандарти безпеки та галузеві сертифікати, але ризик витоку все одно залишається. Перший, хто отримує доступ до цінної комерційної інформації – це постачальник послуг. До того ж йдеться про публічний сервіс, тому питання безпеки не втрачає тут своєї актуальності.

Прив'язка до одного постачальника. При підключенні послуги зазвичай обіцяють гнучку інтеграцію, але коли виникає питання про перехід на хмарний сервіс іншого постачальника, не все йде гладко. Нерідко виникають проблеми щодо сумісності та забезпечення подальшої підтримки [67].

Обмеження контролю для користувача. Повний контроль має лише власник сервісу, клієнту внутрішня інфраструктура сервера недоступна (відкрито можна керувати лише клієнтською частиною застосунків). Тобто кінцевий користувач не може нічого адмініструвати, оновлювати вбудовані програми та керувати ними, до серверної оболонки у нього теж доступу немає.

Основні рівні інфраструктури хмарних технологій:

«Інфраструктура» і «платформа» — це варіанти послуг, що надаються хмарними провайдерами, причому не однакові. В рамках інфраструктури надаються нижче перераховані послуги.

Хмарні сервери. Це є базова послуга. По суті є розміщений у хмарі звичайний комп'ютер з операційною системою та набором засосунків. Сайти та різноманітні програми запускаються саме за допомогою хмарних серверів. Вони можуть мати різні ресурси (мається на увазі потужність, обсяг пам'яті і т.п.). Послугу ще називають VPS/VDS (коли в оренду здається окремий сервер).

Налаштування мережі між хмарними серверами. Зазвичай компанія задіє не єдиний сервер, а кілька. І цей рівень необхідний для того, щоб усі вони могли взаємодіяти як єдине ціле. Тут налаштовуються зв'язки між власними серверами, а також із мережею Інтернет і з серверами, що знаходяться поза хмарою.

Хмарні сховища даних. Тут можна тримати великі обсяги даних, повністю бекапи інфраструктур, що включають множини серверів (зі збереженням налаштувань).

Контейнери. Надається можливість управління контейнерами Docker через Kubernetes, за рахунок контейнерів хмарні програми виходять надійними та максимально стійкими до навантажень.

Визначено, що хмарні технології мають багато як переваг, так і недоліків. З точки зору хмарного провайдера найбільші проблеми виникають при виділенні користувачам хмарних ресурсів.

1.2. Особливості розподілу ресурсів в хмарних обчислювальних системах

На розподіл хмарних ресурсів у відповідній інфраструктурі, суттєво впливає виконання вимог щодо надання ресурсу через Інтернет. У наукових джерелах розглядається багато всіляких методів розподілу ресурсів в обчислювальних та інформаційних структурах. Так, [73] швидкість обробки запиту збільшується за рахунок розпаралелювання обробки. Однак у хмарній

інфраструктурі за деяких технологій це неможливо реалізувати. У статті [65] автором пропонован алгоритм випадкової кластеризації random swap clustering. Слід зауважити, що даний алгоритм суттєво втрачає свою ефективність, якщо його застосовують для хмарної інфраструктури. Характерні особливості хмарних складових системи не розглядаються, якщо проводиться оптимальний розподіл пропускних здатостей для самовідновлювального (англ. self-healing) компоненту комп'ютерної мережі у [88]. У джерелах [74, 108] в основному враховуються тільки характеристики рухомих об'єктів, а не особливості cloud computing. Методи, які пропануються у [79, 102], адаптовані тільки під мобільні компоненти. У статті [106] розглянуті методи для розподілу ресурсів, які наголошуючи на безпекових питаннях, та відсуваються на інший план часові показники,. У статті [48] розглянуто лише одноканальні системи масового обслуговування. Праці [78, 95] спрямовані лише на специфіку Big Data. У статтях [77, 120] розглядаються лише гіперконвергентні системи. Статті [84, 116] орієнтованіа тільки на системи, котрі обробляють відео. А у статтях [55, 122] акцент зроблено лише на оригінальні застосунки.

Однак методи та підходи до розподілу ресурсів, що розглянуті, не враховують багато з особливостей хмарних інфраструктур, що може призвести до непродуктивного використання хмарної інфраструктури. З огляду на це треба чітко визначити характерні риси хмарних обчислень, враховувати які буде необхідно при розподілі хмарних ресурсів. Також необхідно буде виділити технології надання послуг у хмарному середовищі, котрі будуть найбільш вразливими, якщо розподіл або перерозподіл ресурсів проводити неоптимальним чином.

На теперішний час технології, котрі використовують при наданні хмарних послуг, можна поділити на типи, які окреслюються відповідними моделями послуг, що надаються у хмарному середовищі:

- Infrastructure as a Service (інфраструктура у якості послуги, IaaS);
- Platform as a Service (платформа у якості послуги, PaaS);

- Software as a Service (програмне забезпечення у якості послуги, SaaS);
- Calculations as a Service (розрахункові обчислення у якості послуги, сервіс CaaS).

Споживач отримує інформаційно-технологічні ресурси, до яких відносяться віртуальні сервери, що мають певні обчислювальну потужність та обсяги пам'яті, тобто отримують повне Software та віртуальне Hardware, за технологією на базі застосування моделі обслуговування IaaS. Забезпечивши фізичне Hardware та встановивши на нього програмне забезпечення створення віртуальних машин провайдер після цього не займається установкою та підтримкою програмного забезпечення користувача. Провайдером контролюється тільки фізична та віртуальна інфраструктура, яку він надає клієнтові у вигляді віртуальномого середовища за допомогою хмарних серверів. Для цього використовується панель керування, наприклад, цифрова платформа Vmware. Клієнти повністю контролюють всю інфраструктуру і можуть настроїти її під потреби організації. Клієнтами IaaS , зазвичай,стають системні адміністратори компаній. Відомі компанії, які є прикладами IaaS: IBM Softlayer, Hetzner Cloud, Microsoft Azure, Amazon EC2, GigaCloud, Cisco Metacloud.

У разі застосування технології, що використовує модель обслуговування PaaS, провайдер хмари надає користувачам доступ як до операційних систем, так і до програмних засобів розробки та тестування, а також до систем управління базами даних, отже тільки до прикладного та системного Software. Провайдер при цьому контролює не тільки хмарні сервери, хмарні системи для зберігання даних та хмарні обчислювальні потужності, але може запропонувати користувачу певні платформи та засоби управління ними. Найчастіше цю технологію використовують користувачі - розробники Software [115]. За допомогою послуг PaaS вони можуть створювати все, від простих мобільних додатків до складного програмного забезпечення для бізнесу. Приклади

технології PaaS: Google App Engine, IBM Bluemix, Microsoft Azure, VM Ware Cloud Foundry.

Найпоширенішою серед користувачів хмарних провайдерів, є модель надання хмарних послуг SaaS. У цій технології провайдер розробляє та обслуговує програми та сервіси, розміщує їх у хмарі та пропонує, через браузер або застосунок користувачу на його гаджеті. Клієнт лише вносити абонплату (або користується сервісом безкоштовно), оновленням та технічною підтримкою програм займається провайдер. SaaS-сервіси можуть надавати місце для зберігання файлів (Dropbox), офісний пакет документів для роботи (Google Doc, Microsoft Office 365), допомагати організовувати фотографії (Flickr) або спілкуватися з іншими людьми (Facebook) та інше Software. Хмарні технології SaaS орієнтовані на звичайних користувачів, наприклад, на користувачів бізнес-застосунків.

CaaS – це технологія надання послуг, яку також надають деякі провайдери хмарних середовищ. При технології CaaS користувач отримає хмарні ресурси для використання власних застосунків. Дану послугу використовують тоді, якщо саме Hardware не може виконати якесь завдання, зважаючи на якісь причини. Розрахункові завдання, що потребують багато годин для отримання необхідного результату, прогнозні моделі фізичних систем, балістичні розрахунки тощо можуть бути прикладом використання технології CaaS.

Відповідно, кожна технологія має свої особливості щодо завантаження, прогнозування та контролю віртуальної та фізичної інфраструктури ХОС. Так рівень доступу, що має користувач хмарних послуг, який графічно зображений на схемі рис. 1.1, на це суттєво впливає.

IaaS є найбільш гнучкою моделлю хмарного сервісу з простим процесом встановлення обладнання. IaaS дозволяє компаніям розширювати обчислювальні ресурси за потреби, а не купувати дуже дороге обладнання, що потрібно для функціонування власної інфраструктури. Вартість при використанні IaaS варіюється і залежить головним чином від потреби клієнта в

ЦП (центральний процесор) і ОЗП (оперативна пам’ять). IaaS також є економічно ефективною моделлю, також через високу масштабованість і автоматизацію хмарних служб.

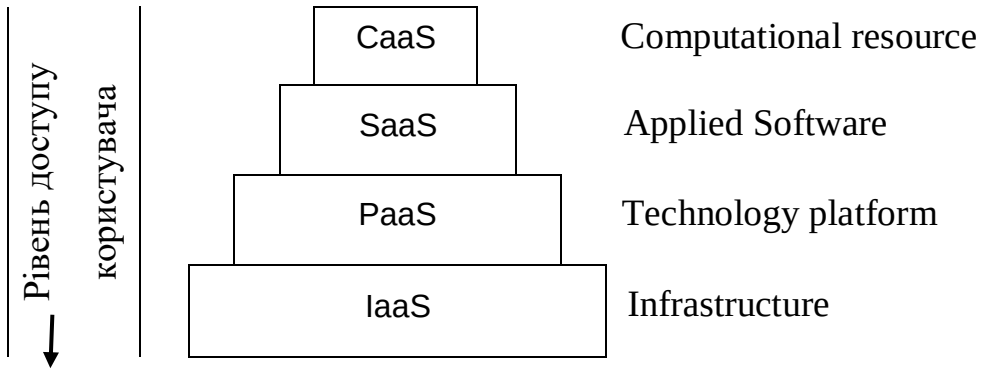


Рисунок 1.1 – Рівні доступу користувачів до хмарних послуг

Сервіси IaaS є актуальними як для багатьох стартапів і малого бізнесу, так і для великих підприємств. Альтернативою закупівлі обладнання та створенню локальної інфраструктури є хмарні сервіси. У міру зростання потреб компанії повинні запроваджувати нові послуги та програми, чому сприяє гнучкість хмарних служб. Таким чином, перехід на IaaS економить як гроші клієнта так і його час.

Але побудувати повністю готове рішення, використовуючи як базу IaaS, за кілька годин практично не можливо. Крім того, якщо у вас є власне обладнання, немає сенсу орендувати віртуальні хмарні пристрої. У цих випадках застосовується платформа PaaS. Як і інші хмарні сервіси, PaaS дозволяє клієнтам використовувати сучасні потужні засоби розробки, підтримує які хмарний провайдер. Також платформа як сервіс хороша тим, що, на відміну від IaaS, відразу готова до роботи. З PaaS збільшується швидкість тестування, розробки та доставки застосунків. На готовій платформі команді розробників буде простіше та економічніше реалізовувати проекти будь-якого розміру та складності. Витрати розгортання платформи і проміжного Software провайдер бере на себе. Це дозволяє, за потреби, динамічно збільшувати або зменшувати

ресурси. Декілька користувачів можуть отримати доступ до проекта через одну і ту ж платформу, яка в свою чергу може працювати з різними веб-службами та базами даних. Тому технологія PaaS може допомогти компаніям будь-якого розміру у завданні оптимізації процесу проведення розробки. Наприклад, використання PaaS може суттєво спростити роботу команди розробників, котрі працюють над одним проектом. Ця технологія найкраще підходить для компаній з наявною IT-інфраструктурою. Клієнтам знадобляться власні IT-фахівці для використання та налаштування програмного забезпечення PaaS-платформи.

Натомість організація отримає більший контроль над процесом розробки та подальшу гнучкість поставки готового додатку клієнтам.

Визначено, багато невеликих компаній не мають можливості купувати готовіі рішення. Зауважимо, що короткострокові проекти, які вимагають простих, швидких та доступних рішень, краще виконувати самій компанії. Для таких випадків найбільш вигідною є технологія SaaS. Віддалене налаштування та обслуговування програмного забезпечення від постачальника дає компанії-замовнику більше часу, котрий можна використати для вирішення будь-яких питань та завдань.

Рішення SaaS управляються централізовано та зберігаються на віддаленому сервері. Відповідальність за встановлення необхідного технічного обладнання і програмного забезпечення у цьому випадку несе виробник, а не користувач. Робота SaaS зазвичай не вимагає завантаження та встановлення програмного забезпечення на пристрої. Більшість програм запускаються у браузері.

Також послуги цієї моделі підходять клієнтам, яким потрібна програма, доступна через інтернет, у тому числі з мобільного пристрою.

Доведено, що у випадках, коли потрібен тимчасовий хмарний сервіс, технологія CaaS є найбільш прийнятною. Його використання дозволяє користувачеві протягом певного періоду поповнювати власні ресурси, набуваючи необхідні віртуальні ресурси.

Тому кожна модель хмари може пропонувати певні можливості та функції. Коли компанія розуміє, які переваги мають різні типи хмарних сервісів та має набір конкретних завдань, то легше вибрати правильний. Технологія IaaS забезпечує майже повний контроль для розробленої віртуальної інфраструктури. Це дозволяє замовнику сформувати стек технологій, котрий буде повністю адаптованим до специфічних потреб бізнесу. Компанії, які вже мають IT-відділ та певні ресурси, також можуть вибрати таку технологію, як PaaS.

Сформована платформа буде в змозі допомагати компаніям при розробці індивідуальних рішень, котрі можна легко інтегрувати в існуючі робочі процеси. Технологія SaaS дозволяє компаніям економити. Клієнтам не потрібно самостійно розробляти та підтримувати програмне забезпечення. А за відсутності ресурсів технологія CaaS незамінна.

На перший погляд розглянуті платформи є схожими, але SaaS, PaaS і IaaS надають послуги різних рівнів. Однак у будь-якому випадку, хмарні рішення знімають роботу з клієнтів та допомагають економити час, зусилля співробітників та гроші. При цьому хмарний провайдер зацікавлений в оптимальному використанні наданих ресурсів.

Це завдання повністю залежить від методів розподілу та перерозподілу ресурсів що використовуються.

1.3. Методи розподілу ресурсів у віртуальних середовищах

На відміну від ХОВ, віртуальні середовища з одного боку, часто мають розмиті кордони між адміністраторами середовища віртуалізації та адміністраторами застосунків, а з іншого боку, забезпечують гнучкість, якщо їх порівнювати з традиційними інфраструктурами, за рахунок як збільшення щільності застосунків, так і відсутності прив'язки до фізичного сервера віртуальних серверів [111, 112]. Але при розподілу ресурсів у ХОВ виникає багато специфічних проблем, які класифіковані на рис. 1.2.

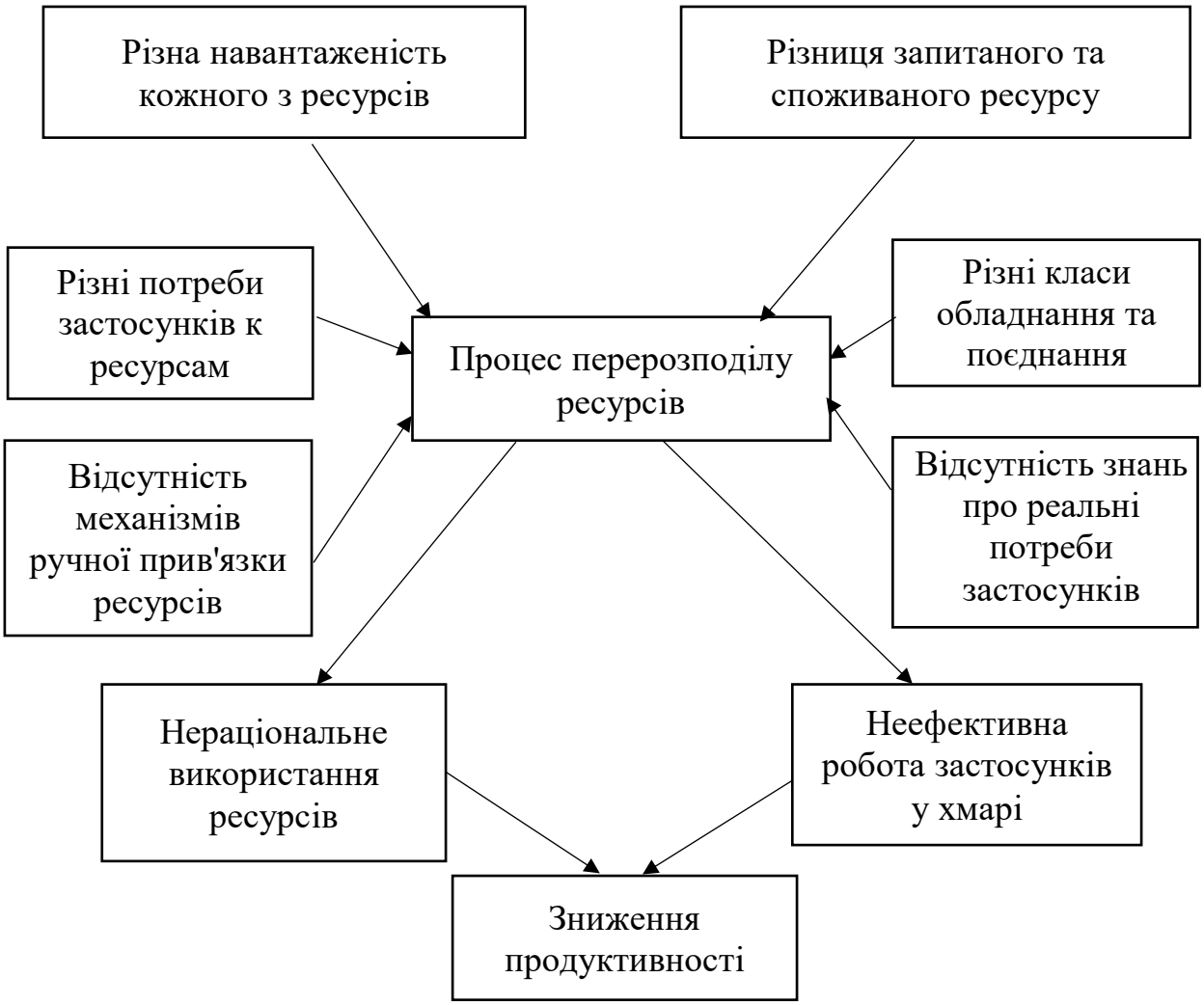


Рисунок 1.2 – Сукупність проблем при розподілі ресурсів у ХОВ

1.3.1. Динамічні методи розподілу базового навантаження

У хмарних інфраструктурах, як і в середовищі віртуалізації, для вирішення проблем планування та оптимізації зазвичай для розподілу та перерозподілу ресурсів використовують такі способи [5]:

• адміністратор, який керує середовищем віртуалізації, має можливість вручну проводити перерозподіл ресурсів;

• планування ресурсів можна проводити автоматично за допомогою методів DPM (Distributed Power Management, розподілене управління

енергоспоживанням) та DRS (Dirstibuted Resource Scheduling, розподілене виділення ресурсів).

Алгоритми методів DRS та DPM відрізняються для різних середовищ, але логіка та принципи роботи зберігаються.

Для методу DRS алгоритм роботи зазвичай складається із низки таких простих кроків [113]:

• обирається хост, найменш навантажений серед тих, що мають достатній ресурс;

• статистика завантаження CPU збирається під час роботи хостів;

• якщо на протязі певного інтервалу CPU на одному з хостів навантаження перевищує встановлений поріг, тоді треба приймати рішення щодо перерозподілу ресурсів;

• якщо прийнято рішення щодо перерозподілу ресурсів, то при використанні методу DRS або здійснюється перенос завдань на інші сервери, або перенос завдання на менш навантажений або на більш продуктивний хост.

З допомогою роботи методу DRS оптимально розподіляється навантаження між серверами, що належать до хмарного середовища [2]. З рахунок цього вдається підвищити як ефективність використання хмарних ресурсів так і пікову продуктивність хостів (рис. 1.3).

Однак метод лише оцінює навантаження на CPU та не може прогнозувати зміну навантаження, розглядаючи інші параметри. З цієї причини метод не часто використовується в хмарних середовищах, у яких адміністратором хмарного середовища не можуть бути передбаченіи всі можливі наслідки перерозподілу ресурсів при використанні даного методу.

Визначено, що метод DPM однозначно вирішує задачу зниження енергоспоживання, але зовсім не вирішує завдання максимальної ефективності використання інших ресурсів [5].

При використанні методу DPM оптимізується енергоспоживання лише на рівні кластера чи вузла. Метод DPM починає виконання з порівняння ресурсів вузла та кластера з тими потребами, що необхідні для віртуальної машини,

орієнтуючись на попередню статистику тих потреб, відповідно до яких вузли віртуалізації перетворюються на режим очікування (рис. 1.4).

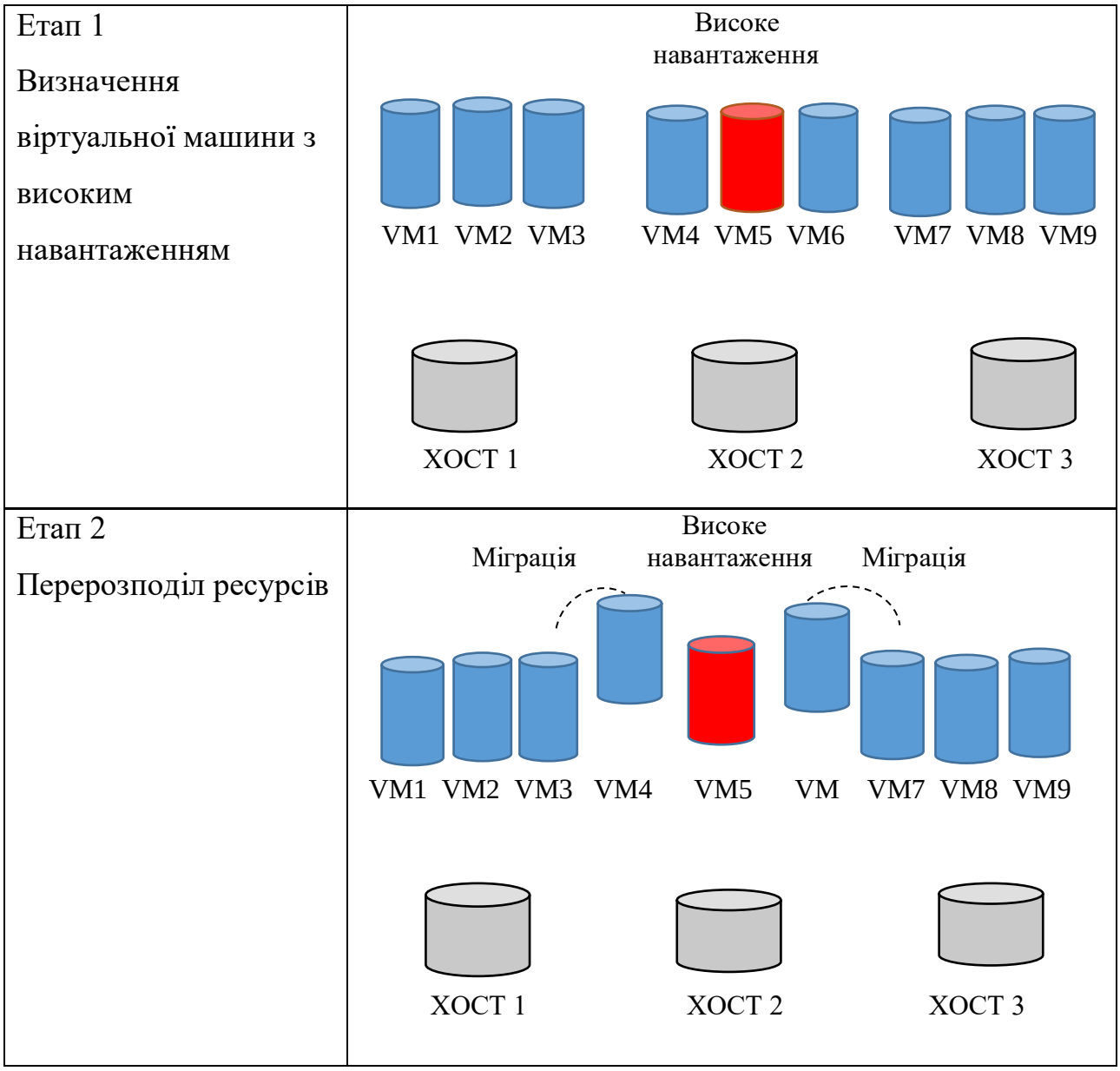


Рисунок 1.3 – Розподіл ресурсів при використанні методу DRS

Якщо зростають потреби в ресурсах, то метод DPM починає запускати вільні вузли та підключати до цих вузлів додаткове навантаження [7].

Алгоритм роботи методу DPM виглядає таким чином:

1) при ініціалізації віртуальної машини обираються найменш навантажені хости серед тих хостів, що мають задовольняючий запиту об’єм досяжної

оперативної пам'яті і, крім того, використовуються іншими віртуальними машинами;

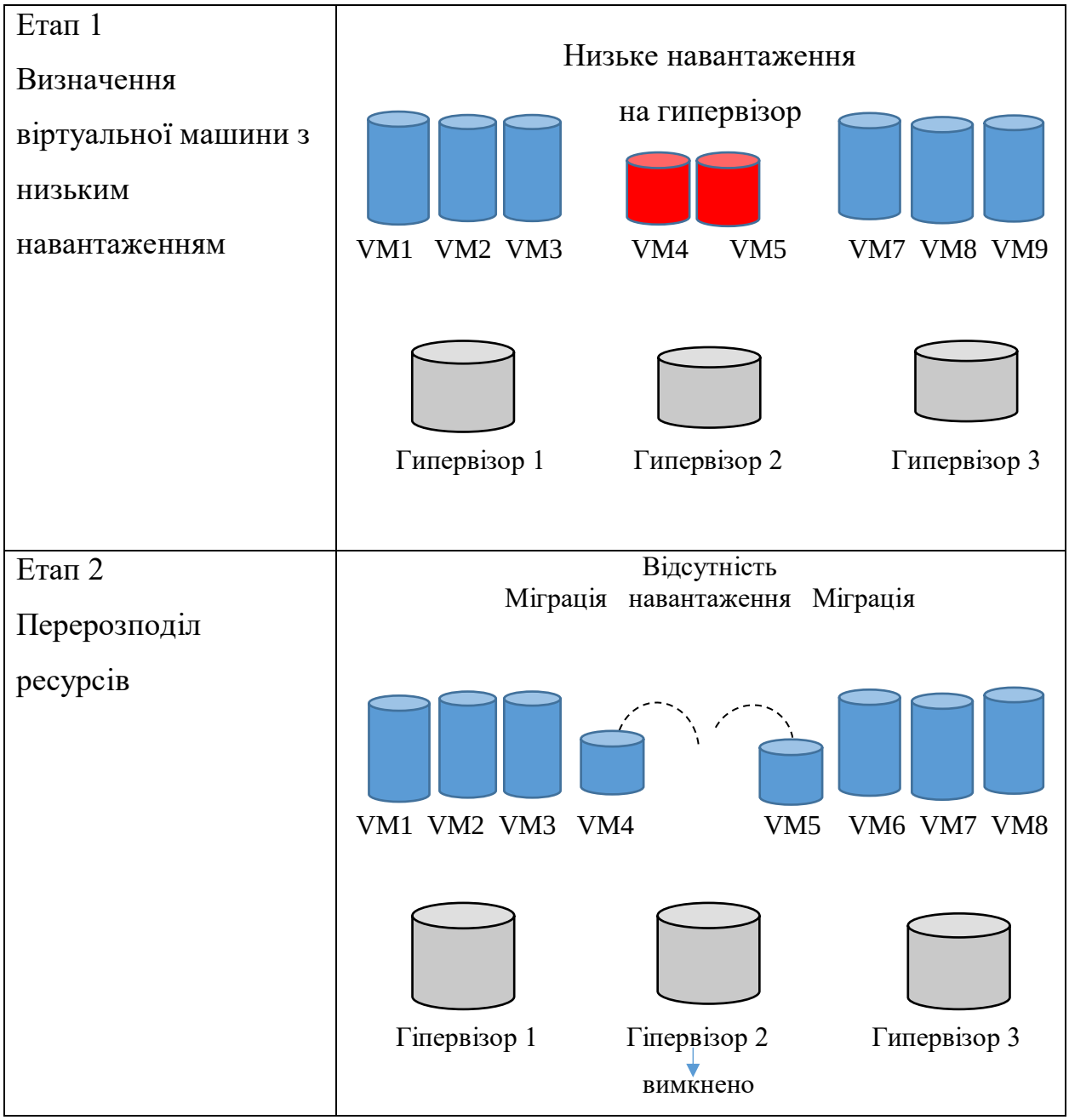


Рисунок. 1.4 – Розподіл ресурсів при використанні методу DPM

2) статистика завантаження ЦП збирається одночасно з роботою хостів;

3) якщо на одному з хостів центральний процесор на протязі фіксованого інтервалу, наприклад, це може бути 5 хвилин, навантажений менше ніж встановлений поріг у відсотках, то приймається рішення щодо необхідності перерозподілу хмарних ресурсів;

4) при прийманні рішення щодо перерозподілу ресурсів, метод DPM здійснює перенос з малонавантаженого хоста до одного з хостів, що на цей момент є найбільш навантаженими;

5) хост, що був звільненим, становиться неактивним та на ньому вимикається енергоживлення до того часу, поки він не буде задіяним, тоді він буде включеним з використанням технології Wake-on-LAN.

У результаті метод DPM може доволі ефективно вирішувати задачу зниження енергоспоживання у хмарному середовищі, проте цей метод ніяким чином не може вирішувати задачі поліпшення використання хмарних ресурсів.

Для вирішення завдань динамічного розподілу ресурсів запропоновано використання методу «інтелектуальних агентів».

Інтелектуальний агент – елемент програми, що самостійно виконує певне завдання, так званий персональний помічник користувача.

В даний час робота цих персональних помічників обмежується простими завданнями. У роботі агент застосовується з метою оцінки роботи алгоритму з погляду швидкодії, тобто часу роботи. Це націлено на те, щоб допомогти користувачеві вибрати найшвидший варіант вирішення поставленої задачі розподілу ресурсів.

Існує кілька типів інтелектуальних агентів:

− агенти з простою поведінкою – діють за схемою «умова – дія» (if – else);

− агенти з поведінкою, заснованою на моделі – оперують із середовищем, яке частково піддається спостереженню;

− цілеспрямовані агенти – зберігають інформацію про ті ситуації, які найбільш бажані для поставленого завдання;

− практичні агенти – розрізняють лише стани, коли мети досягнуто і коли мети не досягнуто, а також розрізняють, наскільки успішний поточний стан;

− учні агенти – незалежні агенти, здатні до навчання і пристосування до обставин, що змінюються.

Розміщення програми «на хмарі» дозволяє мінімізувати витрати підприємства на утримання інфраструктури, що полегшує роботу розробників

щодо внесення змін або розширення можливостей програмного продукту, зручно одночасної роботи з пакетом багатьох користувачів.

1.3.2. Статичні методи розподілу базового навантаження

Для більш ефективного використання ресурсів віртуальних машин є багато способів розподілу базового навантаження у хмарі IaaS провайдерів. Розглянемо докладно три найпопулярніші статичні методи: Round-Robin, Central Manager Algorithm, Threshold Algorithm.

Статичні методи розподілу – це методи, в яких, при розподілу трафіку не враховується стан окремих вузлів. Заздалегідь прописується інформація про параметри вузлів із прив'язкою до певного серверу.

Сервер обирається або випадково, або вибір може бути зумовлений різними факторами, наприклад географічним розташуванням клієнта, що дуже корисно для сайтів із високим розкидом географії відвідувачів.

Round-Robin – алгоритм, який розподіляє навантаження рівномірно між усіма вузлами. При цьому в алгоритмі не враховуються потреби для виконання завдань: обирається першій вузол випадково, а далі по черзі решта вузлів. Коли кількість серверів закінчується, черга знову повертається до першого.

Одним з прикладів алгоритму Round-Robin є Round-Robin DNS. Базова структура Round-Robin DNS зображана на рис. 1.5.

У цьому випадку клієнти отримують різні IP адрси (список із кількох адрес). Це дозволяє провести розподілення навантаження між кількома серверами, які обслуговують програму.

Round-Robin DNS передбачає, що IP адреси будуть видаватися по черзі (спочатку перший, потім другий і т.п.).

Round-Robin DNS в більшості випадків реалізується за допомогою допомогою зворотних проксі, haproxy, apache та nginx. Програма-посередник приймає всі вхідні повідомлення від зовнішніх клієнтів, підтримує список серверів та слідкує за трансляцією запитів.

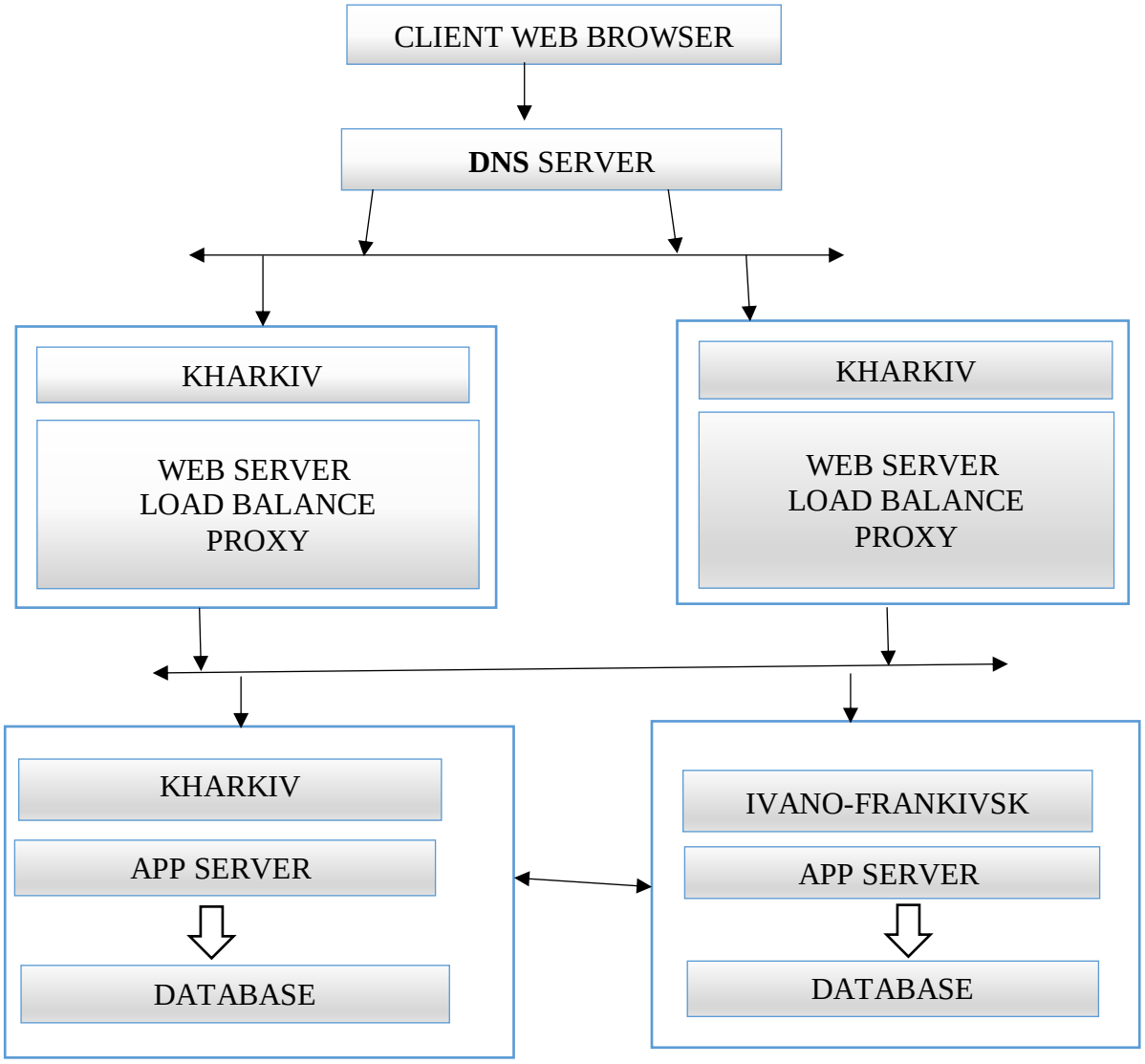


Рисунок 1.5 – Приклад базової структури Round-Robin DNS

Такий підхід має недоліки. Наприклад, DNS не перевіряє сервери на наявність помилок і не виключає зі списку IP-адрес ідентифікатори відключених ВМ. Тому, якщо один із серверів недоступний, можуть виникати затримки в обробці запитів (порядку 10-30 секунд).

IP-адреси доменів кешуються в локальних DNS'ах. При виході з ладу одного з серверів буде тривала затримка перед тим, як оновляться всі кеші. Тому схема Round Robin має використовуватися спільно з механізмами забезпечення стійкості до відмови.

Інша проблема – потрібно підлаштовувати час життя кеша таблиці з адресами. Якщо значення буде надто великим, клієнти не дізнаються про зміни

в групі серверів, а якщо надто маленьким – серйозно зросте навантаження на DNS-сервер.

Central Manager Algorithm – алгоритм в якому центральний обробник (ЦО) визначає хост, нового процесу, вибираючи найменш завантажений сервер. Центральний обробник робить вибір на підставі інформації, яку йому надсилають сервери щоразу при зміні кількості завдань, що обробляються, наприклад, при створенні дочірнього процесу або його припинення (рис. 1.6).

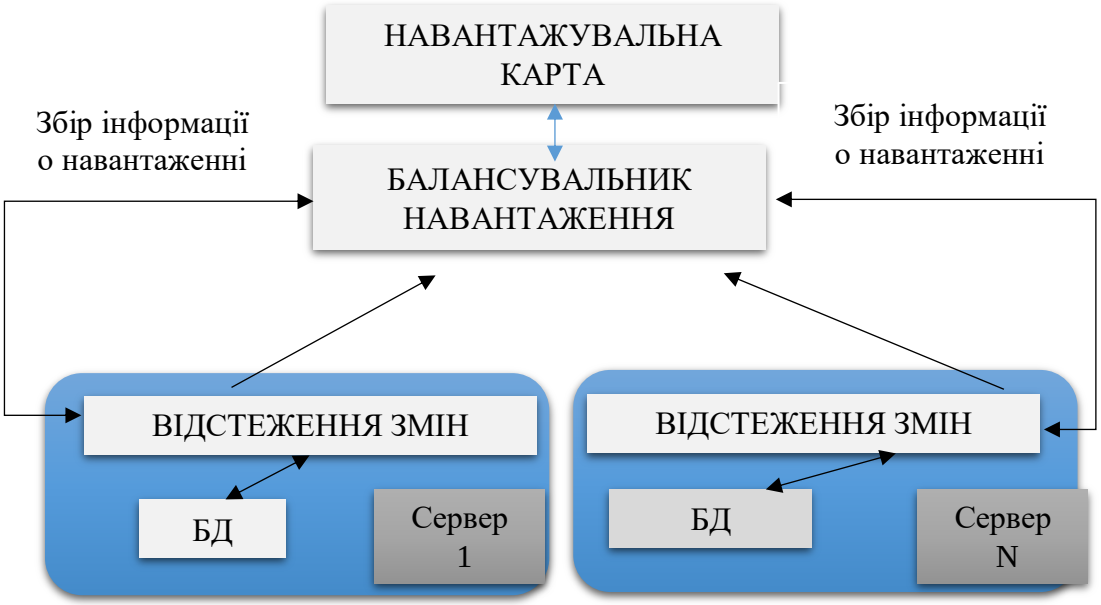


Рисунок 1.6 – Базова структура *Central Manager Algorithm*

Подібний підхід використовується IBM у рішенні Guardium [114, 119]. Додаток збирає та постійно оновлює інформацію про всі керовані модулі та створює на її основі так звану навантажувальну карту. З її допомогою і виконується керування потоками даних. Балансувальник приймає HTTPS-запити від S-TAP – інструмента моніторингу трафіку – прослуховуючи порт 8443 та використовуючи Transport Layer Security (TLS).

Central Manager Algorithm дозволяє розподіляти навантаження, оскільки рішення про призначення процесу серверу виконується в момент його створення.

Однак і цей підхід має серйозний недолік - це велика кількість міжпроцесних взаємозав'язків, що призводить до виникнення «пляшкової шийки». При цьому сам ЦО є єдиною точкою відмови.

Threshold Algorithm. Процеси призначаються хостам відразу при їх створенні. При цьому сервер знаходиться в одному з трьох станів, що визначаються двома пороговими величинами - t_upper і t_under:

- Не завантажено: навантаження < t_under;
- Збалансований: t_under ≤ навантаження ≤ t_upper;
- Перевантажено: навантаження > t_upper.

Якщо система перебуває в збалансованому стані, то ніякі додаткові дії не робляться. Якщо спостерігається дисбаланс, то система робить одне з двох дій: посилає запит збільшення навантаження чи, навпаки, її зниження.

У першому випадку центральний сервер оцінює, які завдання, які ще почали виконуватися іншими хостами, можна делегувати вузлу.

У другому випадку центральний сервер може забрати у вузла ті завдання, які той ще не почав виконувати, та по можливості передати їх іншому серверу.

Під час перерозподіул навантаження, сервери пересилають повідомлення про зміну свого лічильника процесів іншим хостам, щоб оновили інформацію про стан системи.

Перевагою запропанованого підходу є те, що обмін подібними повідомленнями відбувається рідко, оскільки при достатній кількості ресурсів сервера він просто запускає новий процес в собі – це підвищує продуктивність.

1.4. Порівняльний аналіз підходів щодо розподілу ресурсів у хмарному середовищі з різними моделями обслуговування

Отже існуючі методи не дозволяють провести оптимальний розподіл або перерозподіл хмарних ресурсів.

Слід зауважити, що якість проведення розподілу ресурсів буде суттєво залежати від технології, що використовується.

Для порівняння розглянуті основні технології, котрі використовують при наданні послуг у хмарному середовищі:

- інфраструктура у якості послуги, IaaS;
- платформа у якості послуги, PaaS;
- програмне забезпечення у якості послуги, SaaS;
- розрахункові обчислення у якості послуги, сервіс CaaS.

У табл. 1.1 наведені результати проведеного порівняльного аналізу щодо можливості проведення оптимізації по деяких суттєвих параметрах.

Таблиця 1.1− Порівняльний аналіз хмарних технологій

Технологія / **Параметр**	**IaaS**	**PaaS**	**SaaS**	**CaaS**
Вартість ініціалізації	←	←	←	←
Витрати користувача (обслуговування)	←	←	←	←
Витрати провайдера (обслуговування)	←	←	←	←
Перерозподіл ресурсів	–	←	←	←
Прогнозування запитів на ресурси	–	←	←	+
Балансування навантаження	–	←	+	+
Оцінка надійності інфраструктури	–	+	+	+

Згідно з результатами проведеного порівняльного аналізу для хмарних технологій, можна зробити висновок, що методи, які використовуються в хмарному середовищі для реалізації основних технологій обслуговування, у низці випадків не мають можливості перерозподілити хмарні ресурси, витримуючи вимоги до балансування навантаження та здійснити короткочасне прогнозування завантаженості компонентів хмари. Дані питання більш ефективно вирішуються при використанні найпростішої технології CaaS. Хоча немає значних проблем із використанням складних технологій SaaS і PaaS, є кілька функцій, які забороняють використання методів DRS і DPM у деяких сценаріях.

Однак у хмарному середовищі, заснованому на використанні технології IaaS, зазвичай виникають проблеми з оптимальним плануванням ресурсів як на етапі підготовки, так і при розгортанні. Ці проблеми не можна вирішити вручну, це можна зробити за допомогою простих прийомів.

По-перше, адміністратор хмарної інфраструктури не знає, чи потрібно ефективно використовувати програми. По-друге, наявність «статичної» платформи є перешкодою — вона не реагує на потреби, що змінюються з часом.

Тому технологія IaaS не вирішує проблему оптимального розподілу хмарних ресурсів у ХОС. Так, наприклад, одне з найпопулярніших рішень для управління ХОС OpenStack може підтримувати всього три методи для розподілу ресурсів: випадковий метод, випадковий метод у зоні доступності і простий (розподіл ресурсів по черзі). Через нерівномірний розподіл навантаження такі підходи недоступні в хмарному середовищі.

Простий метод розподілу розподіляє ресурси для кожного хоста, задовольняючи запити клієнтів по одному, поки вони не закінчуться. (рис. 1.7).

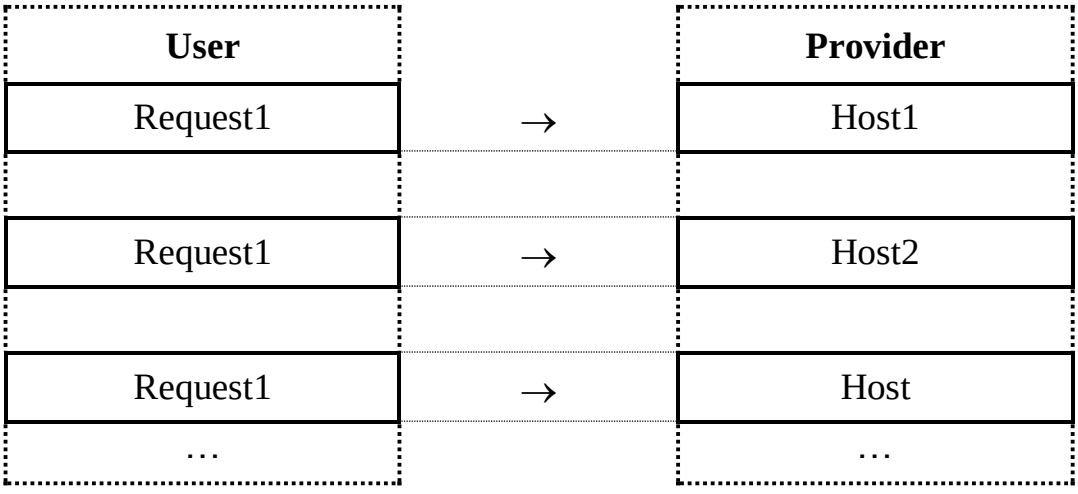


Рисунок 1.7 – Простий метод розподілу ресурсів

При використанні для розподілу ресурсів випадкового методу хост для запуска екземпляра обирається теж випадково між хостів, на котрих є необхідна кількість ресурсів, що необхідна для реалізації запиту (рис. 1.8).

При використанні методу розподілу, що є випадковим у границях зони доступності, діє той же самий принцип, що й при попередньому методі, але будуть задіяними тільки ті хости, що визначені адміністратором заздалегідь у ХОС у якості зони доступності для цього екземпляра (рис. 1.9).

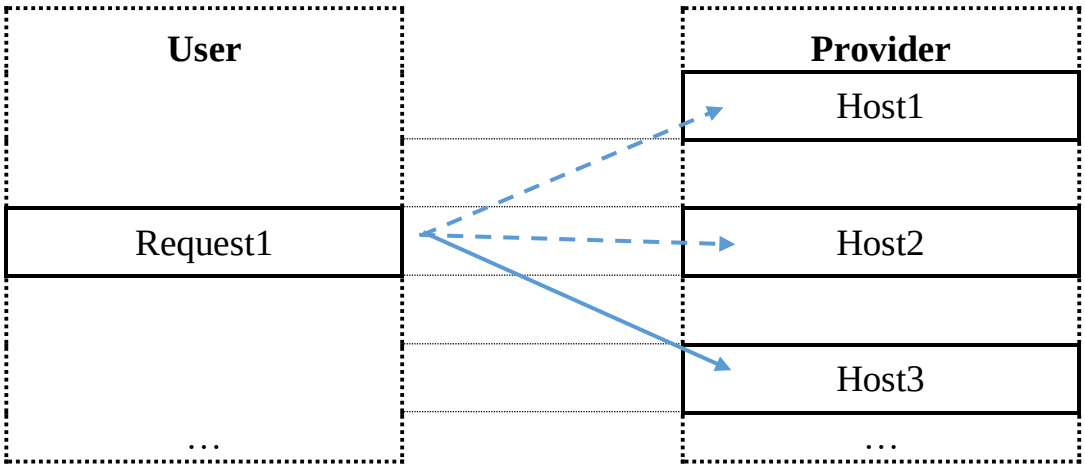


Рисунок 1.8 – Випадковий метод розподілу ресурсів

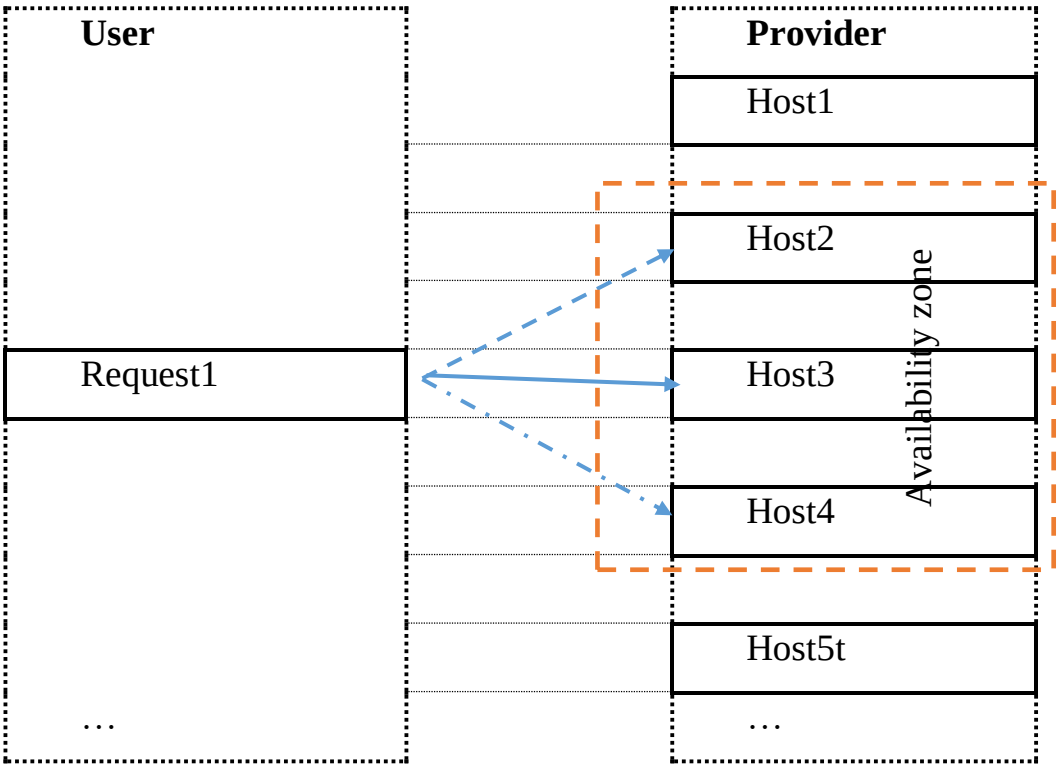


Рисунок 1.9 – Випадковий метод розподілу ресурсів
у межах зони доступності

Таким чином, проведений порівняльний аналіз показав таке: всі три вищенаведені методи для розподілу ресурсів мають при застосуванні у ХОС два суттєві недоліки. Перший полягає в тому, що при прийнятті рішення про використання того чи іншого ресурсу не оцінюється оптимальність рішення. Іншим недоліком є те, що ці методи працюють лише на початковому етапі розподілу ресурсів екземпляра. Крім того, жоден з методів для розподілу ресурсів комп'ютера не враховує такі важливі компоненти, як дискова підсистема і мережа. Тому програми з високими вимогами щодо продуктивності фізичних сховищ (бази даних, системи аналізу тощо) і продуктивності саме мережі (інтернет-шлюзи, відеоконференції тощо) виключаються з логіки при розподілі ресурсів.

Тому, залежно від характеристик хмарного середовища, при розподілі ресурсів у технології IaaS необхідно враховувати розширену низку показників, серед яких є ті, що змінюються у часі:

• продуктивність (швидкість роботи розглянутої мережі, кількість процесорів; кількість ядер у цих процесорів; тактова частота цих процесорів тощо);

• навантаженість (завантаження CPU; зайняте RAM, навантаження на підсистему обслуговування зовнішньої пам’яті; час відгук цієї підсистеми при завантаженні мережі; число екземплярів, що виконуються тощо);

• надійність (фізичний стан поточного обладнання; середній час від відмови до відмови; якісні характеристики обладнання; компетенції обслуговуючого персоналу тощо).

1.5. Обгрунтування завдання підвищення ефективності використання ресурсів хмарного середовища

Отже, розгляд існуючих технологій, що реалізують надання послуг у ХОС, для кожної з котрих визначені переваги, недоліки та характерні особливості, які треба враховувати, розподіляючи ресурси, показав, що

недоліком для кожної з вищеперерахованих технологій є таке, що потреби застосунків, що працюють всередині екземплярів, враховуються тільки в контексті необхідних об’ємів оперативної пам’яті, процесорного ресурсу та об’ємів для даних у сховищах даних. Але при цьому зовсім не враховуються специфічні параметри функціонування застосунків у хмарі, а також те, як з цим застосунком будуть розділяти ресурси інші застосунки, які вже є розгорнутими на тих же хостах. Таким чином, не завжди є місце вибору необхідного ресурсу для проведення розміщення заявок користувачів, що у низці випадків призводить до суттєвого зниження якості роботи застосунків та ефективності використання ресурсів ХОС. Задача складається у тому, щоб провести забезпечення рівномірного розподілу навантажень на всі сервери ХОС різної продуктивності, при цьому надавши кращі умови для функціонування застосунків і в той же час максимально ефективне використовувати наявні обчислювальні ресурси. Показано, що технологія IaaS є найбільш вразливою при нераціональному розподілі або перерозподілі ресурсів.

Отже, доцільно розв’язати актуальну науково-технічну задачу підвищення ефективності розподілу ресурсів у ХОС для мінімізації витрат при використанні технології, що базується на моделі обслуговування «Інфраструктура у якості сервісу» шляхом розробки відповідних моделей та методів, для чого необхідно розглянути та запропонувати рішення таких часткових задач дослідження:

– розробити метод базового виділення ресурсів ХОС користувачу, орієнтованому на модель обслуговування «Інфраструктура у якості сервісу»;

– розробити метод превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель обслуговування «Інфраструктура у якості сервісу»;

– розробити метод адаптивного розподілу ресурсів хмарного середовища;

– провести порівняльну оцінку розроблених та існуючих методів розподілу ресурсів хмарного середовища при використанні технології, орієнтованої на модель обслуговування «Інфраструктура у якості сервісу»;

– дослідити та впровадити розроблені методи розподілу ресурсів хмарного середовища.

Рішення поставлених задач дозволяє підвищити ефективність при використанні ресурсів ХОС в цілому, а також підвищити продуктивність застосунків та знизити поточні витрати на інфраструктуру ХОС та її забезпечення.

1.6. Висновки за розділом 1

Проаналізовані особливості хмарних обчислювальних систем. Окремо розглянуті переваги і недоліки таких систем. На основі аналізу сформульовані загальні вимоги до розвитку обслуговуючого забезпечення хмарних обчислювальних систем.

Розглянуті особливості розподілу ресурсів в хмарному середовищі. Проведений аналіз показав необхідність подальшого вдосконалення процесу розподілу ресурсів.

Розглянуті основні методи розподілу ресурсів у віртуальних середовищах. Зокрема детально виділені характеристики існуючих статичних і динамічних методів, розглянуті підходи до розподілу базового навантаження.

Проведений порівняльний аналіз підходів до розподілу ресурсів у ХОС з різними моделями обслуговування. Показано, що найбільші проблеми виникають при використанні моделі обслуговування «Інфраструктура у якості сервісу».

Сформульована постановка завдання для розв’язання актуальної наукової задачі розробки методів розподілу ресурсів у хмарному середовищі для підвищення ефективності їх використання та мінімізації витрат при використанні технології, що базується на моделі «Інфраструктура у якості сервісу» та виділені основні задачі дослідження.

Основні результати розділу надруковані в наукових працях [1, 9, 12, 14] (додаток А).

РОЗДІЛ 2
РОЗРОБКА МЕТОДУ БАЗОВОГО ВИДІЛЕННЯ РЕСУРСІВ ХМАРНОГО СЕРЕДОВИЩА З МОДЕЛЛЮ ОБСЛУГОВУВАННЯ «ІНФРАСТРУКТУРА У ЯКОСТІ СЕРВІСУ»

При використанні моделі обслуговування «Інфраструктура у якості сервісу» (IaaS**)** клієнт отримує обчислювальні потужності хмари; з урахуванням цієї інфраструктури будуються програмні рішення. IaaS (англ. Infrastructure-as-a-Service) – модель хмарних обчислень, що включає всі основи: серверну інфраструктуру, комунікації, сховища і т.п. Замість того щоб утримувати власну IT-інфраструктуру, клієнт орендує хмару, а провайдер займається обслуговуванням.

У чому переваги IaaS:

– економія бюджету за рахунок передачі видатків на інфраструктуру провайдеру;

– дані розміщуються в дата-центрі провайдера, де вони захищені системами безпеки на фізичному та програмному рівні;

– просте масштабування зі зростанням компанії.

IaaS – база, де будуються програмні рішення компанії. Провайдер IaaS надає менше послуг, ніж у наступних моделях. Залежно від потреб компанії це може бути плюсом.

Якщо у клієнта є власна стратегія розробки IT-рішень, ця модель надає найбільшу свободу для її реалізації. На основі IaaS розгортають як окремі програми, так і всі IT-інструменти компанії – ERP-системи, термінальні сервери, віддалені офіси, бази даних тощо.

Адміністрація та моніторинг IaaS-системи реалізуються за допомогою віртуального інтерфейсу. У ньому ви керуєте конфігурацією IaaS, додаєте встановлене ПЗ і підключаєте додаткові послуги.

Розробка методу розподілу ресурсів у хмарному середовищі, орієнтованому модель надання IT послуг «Інфраструктура у якості сервісу»

(IaaS), є багатоетапним завданням. Вона складається з двох основних завдань: первісне або базове виділення ресурсів; динамічний перерозподіл ресурсів.

У цьому розділі розглянемо підхід до вирішення першого завдання, а саме розглянемо первісне, базове виділення ресурсів.

Враховуючи неоднорідність хмарного середовища, пропонується це завдання вирішувати у 2 етапи:

етап 1: декомпозиція хмарного середовища на зони, виходячи з визначальних особливостей ресурсів, що надаються в кожній зоні;

етап 2: початкове статичне виділення ресурсів всім хостам, що запускаються у хмарному середовищі.

2.1. Декомпозиція хмарного середовища

На першому етапі хмарне середовище декомпозується на зони, виходячи з визначальних особливостей ресурсів, що надаються в кожній зоні [87]. Кожна з зон включає сервери і ресурси зберігання даних зі подібними характеристиками.

Математична модель процесу декомпозиції хмарного середовища S на n зон при використанні моделі обслуговування IaaS має загальний вигляд:

$$S = \bigcup_{i=1}^{n} S_i, \quad \bigcap_{i=1}^{n} S_i = \varnothing, \tag{2.1}$$

де S_i – i-та зона, яка включає ресурси зі схожими характеристиками, тобто

$$S_i = \left(\bigcup_{j=1}^{n_{ih}} h_{ij} \right) \bigcup \left(\bigcup_{\ell=1}^{n_{id}} d_{i\ell} \right), \tag{2.2}$$

де n_{ih}, n_{id} – кількість серверів та кількість сховищ даних у і-й зоні хмарного

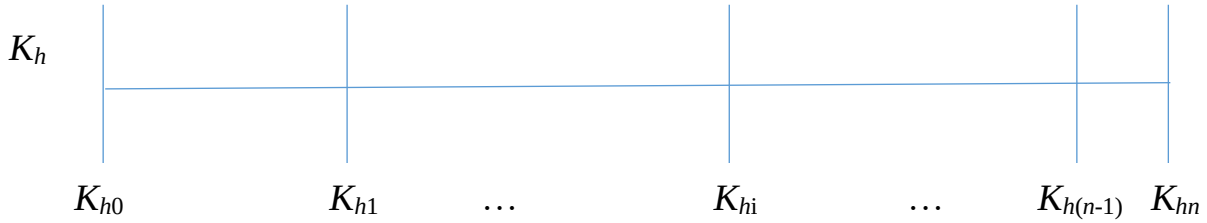
середовища з моделлю обслуговування IaaS відповідно;

$h_{ij},\ j=\overline{1,n_{ih}},\ d_{i\ell},\ \ell=\overline{1,h_{id}}$ – сервери та сховища даних, розташовані в і-й зоні хмарного середовища з моделлю обслуговування IaaS відповідно.

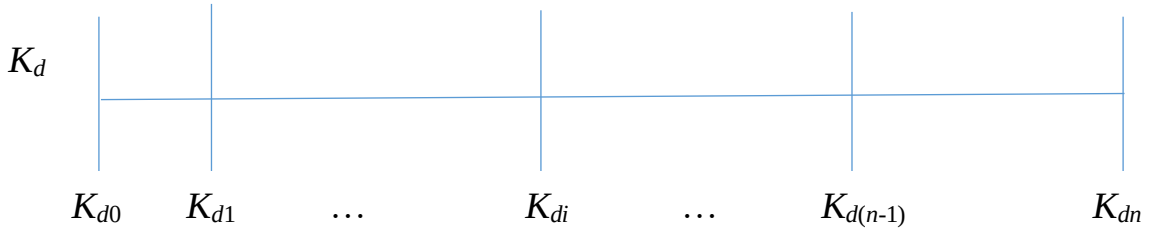
Кожен сервер та кожне сховище даних охарактеризуємо найбільш істотною для даної задачі числовою характеристикою-специфікацією $\left|h_{ij}\right|,\left|d_{i\ell}\right|$ відповідно [89]. Тоді для реалізації процесу декомпозиції хмарного середовища відповідно до виразу (2.1) можна ввести дві числові спадаючі, нерівномірні в загальному випадку, порядкові шкали [96, 103] (рис. 2.1):

$$K_h=\left(K_{h0},\ldots,K_{hi},\ldots,K_{hn}\right),\quad K_{h0}\geq K_{h1}\geq\ldots\geq K_{hi}\geq\ldots\geq K_{hn};\qquad(2.3)$$

$$K_d=\left(K_{d0},\ldots,K_{di},\ldots,K_{dn}\right),\quad K_{d0}\geq K_{d1}\geq\ldots\geq K_{di}\geq\ldots\geq K_{dn}..\qquad(2.4)$$



а



б

Рисунок 2.1 – Вигляд шкал для серверів (a) та сховищ даних (b)

Побудовані шкали (2.3) і (2.4) дозволяють записати необхідні та достатні

умови формування n зон, що не перетинаються, при декомпозиції хмарного середовища з моделлю обслуговування IaaS. Для цього розглянемо випадково вибрані сервер *h* с числовою характеристикою-специфікацією та сховище даних *d* с числовою характеристикою-специфікацією. Тоді:

$$\left(h \in S_i \Leftrightarrow h_{i-1} \le |h| < h_i\right) \& \left(|h| = h_n \Rightarrow h \in S_n\right); \quad (2.5)$$

$$\left(d \in S_i \Leftrightarrow d_{i-1} \le |d| < d_i\right) \& \left(|d| = d_n \Rightarrow d \in S_n\right). \quad (2.6)$$

Вирази (2.5) та (2.6) визначають умови проведення декомпозиції (2.1) та ґрунтуються на шкалах (2.3) та (2.4). У свою чергу, дані шкали формуються адміністратором хмарного середовища, виходячи з його експертних знань про особливості роботи обладнання та його сумісність. Також при цьому може бути враховано передісторію експлуатації даного хмарного середовища з моделлю обслуговування IaaS.

На прикладі представлено процес декомпозиції хмарного середовища на 3 зони. Характеристикою-специфікацією для серверів визначено кількість CPU, а дискових сховищ – тип класу дискового масиву (визначає потенційні можливості використання).

Нехай досліджуване хмарне середовище базується на таких обчислювальних засобах:

– 2 двопроцесорні сервери;

- 2 чотирипроцесорні сервери;

– 3 восьмипроцесорні сервери;

- 1 шістнадцятипроцесорний сервер;

- 2 двадцятичотирипроцесорні сервери;

- 2 дискових масиву HP XP12000 (клас Hi-End);

- 1 дисковий масив Hitachi USP-V (клас Hi-End);

- 2 дискових масиви IBM DS-4700 (клас Mid-Range);

- 3 дискових масиви SDS DS-207 (клас Entry-level).

Виходячи з попереднього аналізу запитів користувачів передбачається побудова таких зон навантаження:

високого (High load zone, HZ),

середнього (Medium load zone, MZ)

низького (Low load zone, LZ).

Виходячи з цього, при запуску віртуальний хост розміщуватиметься в тій зоні, ресурси якої йому, швидше за все, знадобляться, виходячи з початкових характеристик (число CPU, обсяг оперативної пам'яті, обсяг сховища). Для проведення декомпозиції формуються дві шкали:

- чисельна шкала Kh = (24, 8, 4, 2);

- якісна шкала Kd = (Hi-End; Mid-Range, Entry-level, 0).

Результат декомпозиції наочно показано на рис. 2.2.

Після етапу проведення декомпозиції можна здійснити початковий розподіл ресурсів між заявленими віртуальними хостами – базовий розподіл за первісною заявкою користувача хмари. І тому розглянемо методи статичного розподілу ресурсів.

2.2. Обґрунтування вибору методу для базового виділення хмарних ресурсів

При виборі методу статичного розподілу ресурсів слід орієнтуватися на характерні особливості моделі обслуговування IaaS. Аналіз існуючих методів розподілу ресурсів для багатокритеріальних завдань прийняття рішень, проведений у [36, 37, 41], показав можливість використання математичного апарату методу аналізу ієрархій [104].

Основне застосування методу – підтримка прийняття рішень у вигляді ієрархічної композиції завдання та рейтингування альтернативних рішень. Опишемо можливості цього методу [104].

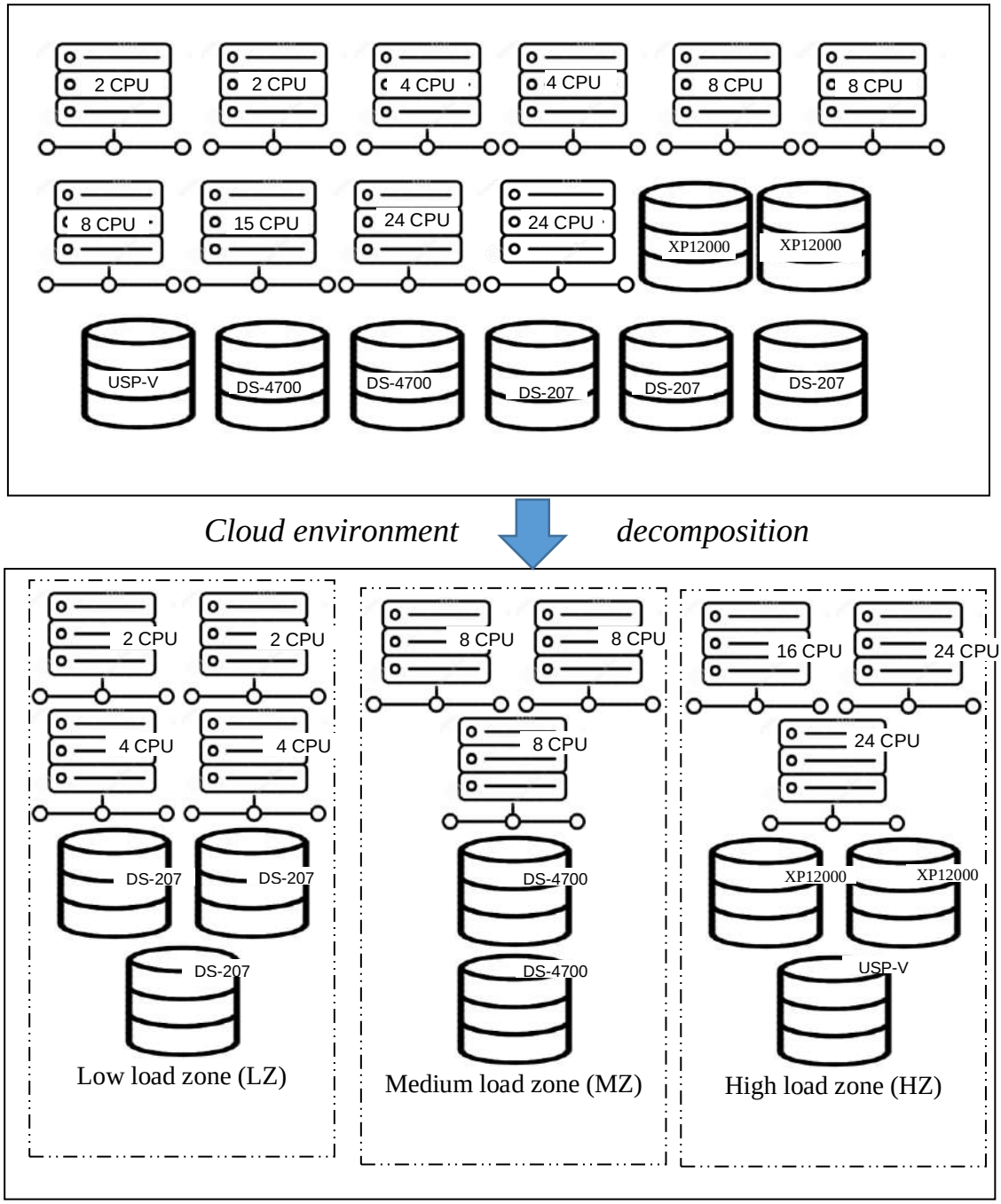


Рисунок 2.2 – Приклад декомпозиції хмарного середовища на 3 зони

- *Метод дозволяє провести аналіз недоліків .В цьому контексті проблема ухвалення рішення представляється в такому вигляді ієрархічно упорядкованих компонентів:*

– основного критерію (головної мети) побудова рейтингу можливих рішень,

– декількох рівнів (груп) факторів одного типу, які мають вплив на рейтинг (так ци інакше),

– невеликої кількості можливих рішень,

– системи зв'язків, як такі, що вказують на взаємний вплив факторів і рішень.

- *Метод дозволяє провести збір даних із проблеми*. Модель ситуації ухвалення рішення, що відповідає результатам ієрархічної декомпозиції, має кластерну структуру. Фактори, які мають вплив на пріоритети рішень, та всі можливі рішення, розбиваємо на кластери (невеликі групи подібних елементів). За допомогою запропонованої процедури попарних порівнянь, яка розроблена методом аналізу ієрархій, з’являється можливість визначення пріоритетів об'єктів, які входять у кожен кластер. І тому використовується метод власного вектора. Отже виникає ряд більш простих проблем збору даних, замість однієї складної проблеми.

- *Метод дозволяє оцінити суперечливість даних та мінімізувати її.* З цією метою розроблено процедуру узгодження всередині аналізу ієрархій. Для виявлення найменш ясних ділянок проблеми та організуватибільш ретельного вибіркового обдумування проблеми, при використанні запропонованого методу є можливість визначати найбільш суперечливі дані

- *Метод дозволяє провести синтез проблеми ухвалення рішення.* Метод дозволяє провести синтез проблеми ухвалення рішення.Набір альтернативних рішень які найбільш пріоритетні, а саме підсумковий рейтинг, розраховується, за спеціальним алгоритмом, після проведення аналізу проблеми та збору усіх даних. Завдяки властивостям цього рейтингу є можливість підтримувати прийняття рішень. За основне, приймається рішення, пріоритет якого найвищий.

Крім того запропонований метод дозволяє оцінити важливість кожного фактору за рахунок можливості побудувати рейтинг групи факторів.

- *Метод дозволяє організувати обговорення проблеми, сприяє*

досягненню консенсусу. У цій ситуації, навіть думки, які можуть виникнути під час обговорення проблеми прийняття рішення мають можливість розглядатися як можливі рішення. Тому, можна визначити, що при використання методу аналізу ієрархій, думка кожного учасника обговорення, є важлививою.

- *Метод дозволяє оцінити важливість обліку кожного рішення та важливість обліку кожного фактору, що впливає на пріоритети рішень.* У відповідності до формулювання завдання з прийняття рішень, оптимальне рішення, безпосередньо, пов'язане саме з величиною пріоритету. І тому, як несуттєві, відкидаються рішення з низькими пріоритетами. . Як зазначено вище, метод дозволяє оцінювати пріоритети факторів. Тому, якщо за винятком деякого чинника пріоритети рішень змінюються незначно, такий чинник вважатимуться несуттєвим для завдання.
- *Метод дозволяє оцінити стійкість прийнятого рішення.* Якщо, неточність моделі ситуації прийняття рішення або неточність даних не впливають істотно на рейтинг рішень, можна вважати обґрунтованим рішення, що приймається.

До переваг запропанованого методу під час вирішення завдання початкового виділення ресурсів у хмарних середовищах, проти інших методів багатокритеріального аналізу слід зарахувати [33]:

- Під час підготовки прийняття решения має місце відсутність загальних правил які б формували структуру моделі прийняття рішення, з врахуванням «людського чинника». Запропонований метод дозволяє работу підсистеми розподілу наблизити до дій системного адміністратора, який може розподілити ресурси між різними параметрами хмарного середовища.
- Процедура розрахунків рейтингів являється простою, і це робить процес прийняття вибору, при налаштуванні підсистеми розподілу ресурсів, максимально прозорим.
- Завдяки можливості враховувати думку не одного, а кількох експертів, яку надає запропонований метод, отримуємо більш об'єктивні

оцінки значущості показників.

- Метод надає великі можливості для виявлення та мінімізації суперечностей даних, і це дає змогу ефективно оперувати великою кількістю показників у процесі ухвалення рішення.
- Метод дає узагальнений підхід, ніж метод логічних ланцюгів, як спосіб виявлення найкращого рішення, так і дозволяє кількісно відобразити ранг переваги за допомогою розстановки рейтингів, показуючи природний хід людського мислення, що дозволяє більш ефективно використовувати отримані результати у разі їх подальшого використання (наприклад, якщо запуск віртуального хоста на сервері, що є найкращою альтернативою, завершується з помилкою).

Метод аналізу ієрархій, тим не менш, має і ряд недоліків, які перераховані нижче.

- Важливий недолік, який частково обмежує можливості застосування методу аналізу ієрархій - відсутнфсть дієвих засобів перевірки даних, при використанні методу.

Однак, в принципі, і не може бути об'єктивних даних при виборі оптимального Hardware для розміщення віртуального хоста в хмарному середовищі через їхню погану формалізацію і неявну залежність. Якість таких даних визнається задовільною, якщо збір даних проведено за допомогою спеціалістів з досвідом, і в даних немає суттєвих протиріч.

- Метод дає лише можливість рейтингувати альтернативи, при цьому, відсутні внутрішні засобіви для інтерпретації рейтингів, тому вважається, що людина, яка приймає рішення, повинна, в залежності від ситуації сама зробити висновок, знаючи рейтинг можливих рішень.

Тим не менш, в рамках підсистеми розподілу ресурсів у хмарному середовищі з моделлю обслуговування IaaS, отриманих даних достатньо для ухвалення рішення. Отже, наведені недоліки методу ієрархій при початковому розподілі ресурсів для завдання не є суттєвими.

Тому можна вважати, доцільність застосування методу аналізу ієрархій

для вирішення завдання початкового розподілу ресурсів у хмарному середовищі з моделлю обслуговування IaaS, обґрунтованою

2.3. Метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель IaaS

Початковий або базовий розподіл ресурсів для кожного віртуального хоста, що запускається, у хмарному середовищі з моделлю обслуговування IaaS здійснюється з використанням методу аналізу ієрархій.

Як відомо, в різних галузях науки і техніки для вирішення подібних завдань, успішно застосовується метод аналізу ієрархій Сааті [28], проте, в силу особливостей розв'язуваного завдання, вимагає виконання низки кроків, що враховують специфіку хмарних середовищ з моделлю обслуговування IaaS [4, 8] (рис. 2.3).

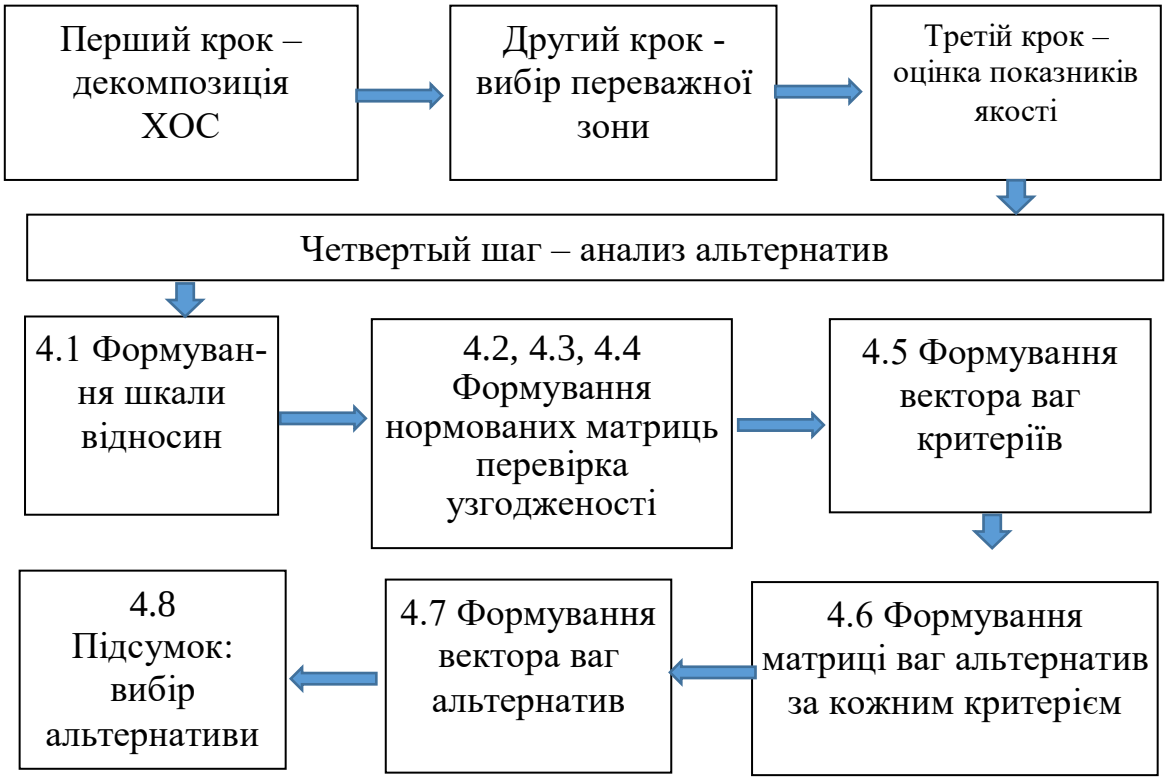


Рисунок 2.3 – Схема застосування методу аналізу ієрархій Сааті

На першому кроці проводиться декомпозиція обчислювального хмарного середовища на зони відповідно до підходу, описаного вище. Характеристики Hardware у кожній із зон близькі одна до одної.

На другому кроці з урахуванням заданих для віртуального хоста обмежень (на кількість процесорів, обсяг пам'яті, ресурси зберігання та ін) вибирається найкраща йому зона. Обмеження екземпляра, що запускається за характеристиками, щоб вибрати його розташування вже там, де ресурсів свідомо буде достатньо для функціонування додатків, що виконуються на віртуальному хості, враховується при використанні зон . У вибраній зоні визначається безліч усіх можливих альтернатив для Hardware (обладнання), що забезпечує функціонування необхідного віртуального хоста:

$$A = \{A_1, \ldots, A_m\}, \quad card\ A = m. \tag{2.7}$$

На третьому етапі оцінюються показники якості, якими описується функціонування кожного компонента Hardware, включеного у вибрану зону. На цьому кроці для кожного хоста із зони, визначеної на попередньому етапі, виконується оцінка показників якості і їх значимість [39, 42], якими описується функціонування Hardware.

Комплексний показник якості визначається безліччю показників якості, що розглядаються. Ця множина складається з трьох підмножин, що являють собою групи основних показників якості для хмарного середовища з моделлю обслуговування IaaS

$$MP = \{P, R, S, C\}, \tag{2.8}$$

де P – підмножина приватних показників продуктивності;

R – підмножина приватних показників надійності;

S - підмножина приватних показників навантаженості;

C – підмножина вартісних показників.

Відповідно, дані для множини складаються з таких часткових показників якості:

$$P = \{ p_1, \dots p_{n1} \}; \tag{2.9}$$

$$R = \{ p_{n1+1}, \dots p_{n2} \}; \tag{2.10}$$

$$S = \{ p_{n2+1}, \dots p_{n3} \}. \tag{2.11}$$

$$C = \{ p_{n3+1}, \dots p_{n} \}. \tag{2.12}$$

де n – загальна кількість часткових показників, серед яких:

n1 – кількість показників продуктивності,

(n2 – n1) – кількість показників надійності,

(n3 – n2) – кількість показників навантаженості,

(n – n3) – кількість вартісних показників.

При цьому, сукупність часткових показників якості, які впливають на показник якості в комплексі, може бути зведено у 2 групи:

детерміновані (ті, що пов'язані зі зміною обладнання);

стохастичні (ті, що пов'язані з навантаженням та часом роботи).

Окремі показники, з вище зазначених, такі, як показники фізичного стану обладнання та рівень компетенції обслуговуючого персоналу, не можуть бути кількісно визначені, а отже є нечіткими та можуть бути представлені у вигляді лінгвістичних змінних [38, 40].

На четвертому кроці для вибору кращого Hardware для розміщення віртуального хоста виконується аналіз альтернатив на основі отриманих

значень показників якості [44]. Кожен показник має свою попередньо задану змачимість, яка визначена експертним шляхом, виходячи з узагальнених потреб клієнтів та наявного обладнання, для кожної зони. В результаті для розміщення віртуального хоста вибирається склад Hardware, що забезпечує максимальну можливість оптимального розподілу ресурсів $Pmax$.

Запропонований алгоритм виконання аналіаза альтернатив на основі отриманих значень показників якості має такі кроки:

Крок 4.1. Попередньо визначається якісна шкала для проведення попарного порівняння з подальшим перетворенням на бали відповідно до табл. 2.1. В термінах домінування одного елемента над іншим проводяться парні порівняння.

Таблиця 2.1 – Шкала відносин для параметрів хмарного середовища

Ступінь значимості	**Визначення**	**Пояснення**
1	Однакова значимість	Удвох параметрів однаковий внесок у досягнення мети
3	Певна перевага значимості однієї дії над іншою (слабка значимість)	Переваги одного з параметрів, проте недостатньо переконливі
5	Істотна чи сильна значимість	Існують надійні дані або логічні міркування для того, щоб показати перевагу одного з параметрів
7	Очевидне або дуже сильне значення	Переконливе свідчення на користь одного параметра перед іншим
9	Абсолютна значимість	Свідоцтва на користь переваги одного параметра іншому переконливі
2, 4, 6, 8	Проміжні значення між двома сусідніми судженнями	Ситуація, коли необхідне компромісне рішення

Якщо параметр *i* при порівнянні з параметром *j* приписується одне з визначених вище ненульових чисел, то параметр *j* при порівнянні з параметром *i* приписується зворотне значення.

Крок 4.2. Формуємо квадратну матрицю попарних порівнянь усіх показників розміром $n \times n$:

$$M_{crit} = \left(\xi_{j\ell}\right), \quad j, \ell = \overline{1,n}. \tag{2.13}$$

Крок 4.3. До кожного показника формуємо квадратну матрицю попарних порівнянь всіх можливих альтернатив, тобто. отримуємо масив матриць:

$$M_{pi} = \left(\eta_{ij\ell}\right), \quad i = \overline{1,n}; \;\; j, \ell = \overline{1,m}. \tag{2.14}$$

Крок 4.4. Нормуємо матриці (2.13), (2.14) за наступною схемою:

$$M_{pi}^{(norm)} = \left(\eta_{ij\ell} \Big/ \sum_{\zeta=1}^{m} \eta_{i\zeta\ell} \right), \quad i = \overline{1,n}; \;\; j, \ell = \overline{1,m}. \tag{2.15}$$

Крок 4.5. З нормованої матриці (2.15) формуємо ваговий стовпець-вектор критеріїв розміром $(1 \times n)$, знаходячи середнє значення для кожного рядка даної матриці:

$$V_{crit}^{(weight)} = \left(v_j\right) = \left(\sum_{\ell=1}^{n} \left(\xi_{j\ell} \Big/ \sum_{\zeta=1}^{n} \xi_{\zeta\ell} \right) \Big/ n \right), \quad j = \overline{1,n}; \tag{2.16}$$

Крок 4.6. З нормованих матриць (2.16) формуємо матрицю ваг альтернатив по кожному критерію (що складається з отриманих аналогічним чином вагових стовпців) розміром $(m \times n)$:

$$M_p^{(weight)} = \left(m_{ij}\right) = \left(\sum_{\ell=1}^{m} \left(\eta_{ij\ell} \Big/ \sum_{\zeta=1}^{m} \eta_{i\zeta\ell} \Big/ m \right) \right), \quad i = \overline{1,n}; \;\; j = \overline{1,m} \qquad (2.17)$$

Крок 4.7. Помножуючи отриману матрицю (2.17) на стовпець (2.16) за правилом множення матриць, отримуємо нормований (за побудовою) вектор ваг альтернатив розміром $(1 \times m)$ з точки зору досягнення мети:

$$W_A = M_p^{(weight)} \cdot V_{crit}^{(weight)} = \left(w_j\right), \quad j = \overline{1,m}; \quad \sum_{j=1}^{m} w_j = 1.. \qquad (2.18)$$

Крок 4.8. Тоді найприйнятнішим рішенням з погляду методу аналізу ієрархій є така альтернатива, що відповідає максимальному елементу отриманого вектора (2.18), тобто.

$$A_{pr} = \left(A_i \middle| \, w_i = \max_j \left(w_j\right), \quad j = \overline{1,m} \right). \qquad (2.19)$$

2.4. Приклад базового завантаження віртуального хоста

Запуск віртуального хоста у хмарному середовищі з моделлю обслуговування IaaS розглянемо на конкретному прикладі.

Як приклад беремо хмарне середовище, розглянуте вище, результат декомпозиції якої (крок 1) наведено на рис. 2. Основні параметри функціонування обчислювального середовища (вирази (2.8) – (2.11)) описуються такими значеннями:

$$P = \left\{ N_{CPU}, N_{Nucl}, V_{CPU}, V_{net} \right\}; \qquad (2.20)$$

$$R = \left\{ \Delta T_{fail}, Cond_{hw}, Serv_{pers}, VM \right\}; \qquad (2.21)$$

$$S = \{V_{RAM}, Z_{CPU}, Z_{net}, EM, N_{vh}\}. \quad (2.22)$$

$$C = \{c_{rent}\}. \quad (2.23)$$

де N_{CPU} – число CPU;

N_{Nucl} – кількість ядер в кожному CPU;

V_{CPU} – тактова частота CPU;

V_{net} – швидкість роботи мережі;

ΔT_{fail} – середній час між збоями;

$Cond_{hw}$ – ступінь фізичного стану Hardware;

$Serv_{pers}$ – рівень компетенцій персоналу для обслуговування обладнання;

VM – середня швидкість доступу до зовнішньої пам'яті;

EM – обсяг доступної зовнішньої пам'яті;

V_{RAM} – вільна область RAM;

Z_{CPU} – завантаження CPU,

Z_{net} – завантаження мережі,

N_{vh} – кількість екземплярів віртуальних хостів;

c_{rent} – відносна вартість місячної оренди.

Для виконання кроку 2 визначимо мінімальні вимоги до Hardware для забезпечення функціонування віртуального хоста:

$$N_{CPU} \geq 12;\ N_{Nucl} \geq 12;\ V_{CPU} \geq 2\ \text{GHz};$$
$$V_{net} \geq 1000\ \text{Mbps};\ \Delta T_{fail} \geq 1000\ \text{hour};\ Cond_{hw} \geq 0.95;\ Serv_{pers} = 1. \quad (2.24)$$

Як очевидно з умов (2.24) для даного віртуального хоста необхідно вибрати зону високої продуктивності (High load zone, HZ, рис. 2.2).

Для виконання третього кроку визначимо поточні значення показників для Hardware із зони HZ (табл. 2.2, 2.3).

Таблиця 2.2 – Характеристики серверів

Hardware	Продуктивність				Надійність			Навантаженість			
	N_{CPU}	N_{Nucl}	V_{CPU}	V_{net}	ΔT_{fail}	$Cond_{hw}$	$Serv_{pers}$	V_{RAM}	Z_{CPU}	Z_{net}	N_{vh}
Server 1	16	12	2 GHz	10^3 Mbps	10^4 h	0,96	1	50 %	60 %	30 %	5
Server 2	24	16	4 GHz	$5 \cdot 10^3$ Mbps	10^4 h	0,96	1	70 %	20 %	30 %	2
Server 3	24	20	8 GHz	$8 \cdot 10^3$ Mbps	10^5 h	0,99	1	60 %	40 %	50 %	3

Таблиця 2.3 – Характеристики дискових масивів

Дискові масиви	*VM*	*EM*
XP12000 - 1	7 Gb/s	40 %
XP12000 - 2	11 Gb/s	60 %
USP-V	25 Gb/s	90 %

Умовам (2.24) задовольняють будь-які варіанти Hardware, тому на виході кроку 3 буде сформовано 9 альтернативних варіантів (табл. 2.4). За одиницю відносної вартості місячної оренди взято вартість для найдешевшої альтернативи.

Таблиця 2.4 – Список альтернатив

№	Hardware	c_{rent}
1.	Server 1, XP12000 - 1	1
2.	Server 1, XP12000 - 2	1.1
3.	Server 1, USP-V	1.5
4.	Server 2, XP12000 - 1	1.2
5.	Server 2, XP12000 - 2	1.3
6.	Server 2, USP-V	1.6
7.	Server 3, XP12000 - 1	1.7
8.	Server 3, XP12000 - 2	1.8
9.	Server 3, USP-V	2.4

Наразі можливий перехід до аналізу альтернатив (крок 4). Якісна шкала щодо попарного порівняння з наступним перетворенням на бали визначається відповідно до табл. 2.1.

Формуються матриці попарних порівнянь всіх показників і всіх можливих альтернатив (рис. 2.4 – рис. 2.13).

	ПРОДУКТИВНІСТЬ				НАДІЙНІСТЬ				НАВАНТАЖЕННІСТЬ					ВАРТІСТЬ	Множення	√ множення	локальний вектор приоритетів
	NCPU	NNUCL	VCPU	VNET	Tfail	Condhw	Servpers	VM	EM	VRAM	ZCPU	Znet	Nvh	Crent			
NCPU	1	1	3	3	3	7	9	1	1	1	1	3	3	3	45927	2,152805839	0,133588411
NNUCL	1	1	1	1	3	1	7	1	1	3	1	1	3	9	1701	1,701236116	0,105567081
VCPU	1/3	1	1	5	7	1	5	3	1	3	1	1	1	7	3675	1,797468295	0,111538592
VNET	1/3	1	1/5	1	1	3	5	3	3	1	1	3	3	7	567	1,572839851	0,097599687
Tfail	1/3	1/3	1/7	1	1	3	7	1	5	1	1	1	3	3	15	1,213407063	0,075295746
Condhw	1/7	1	1	1/3	1/3	1	5	7	1	3	1	3	3	7	105	1,39434635	0,086523601
Servpers	1/9	1/7	1/5	1/5	1/7	1/5	1	1	5	5	7	7	5	7	0,777777778	0,982209129	0,060949183
VM	1	1	1/3	1/3	1	1/7	1	1	1	5	1	3	1	9	2,142857143	1,055947613	0,065524889
EM	1	1	1	1/3	1/5	1	1/5	1	1	3	1	3	1	9	1,08	1,005512355	0,062395222
VRAM	1	1/3	1/3	1	1	1/3	1/5	1/5	1/3	1	1	1	3	3	0,004444444	0,679183244	0,042145468
ZCPU	1	1	1	1	1	1	1/7	1	1	1	1	1	3	9	3,857142857	1,101225155	0,068334503
Znet	1/3	1	1	1/3	1	1/3	1/7	1/3	1/3	1	1	1	1	9	0,005291005	0,687694555	0,042673622
Nvh	1/3	1/3	1	1/3	1/3	1/3	1/5	1	1	1/3	1/3	1	1	7	0,000640146	0,591394239	0,036697883
Crent	1/3	1/9	1/7	1/7	1/3	1/7	1/7	1/9	1/9	1/3	1/9	1/9	1/7	1	3,73193E-11	0,179944305	0,011166113

Рисунок 2.4 – Матриці попарних порівнянь всіх показників

NCPU	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	5	5	5	5	5	5	15625	2,924017738	0,238095238
2	1	1	1	5	5	5	5	5	5	15625	2,924017738	0,238095238
3	1	1	1	5	5	5	5	5	5	15625	2,924017738	0,238095238
4	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
5	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
6	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
7	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
8	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
9	0,2	0,2	0,2	1	1	1	1	1	1	0	0,584803548	0,047619048
											12,2808745	

Рисунок 2.5 – Матриця попарних порівнянь альтернатив N_{CPU}

NNucl	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	3	3	3	5	5	5	3375	2,466212074	0,216109671
2	1	1	1	3	3	3	5	5	5	3375	2,466212074	0,216109671
3	1	1	1	3	3	3	5	5	5	3375	2,466212074	0,216109671
4	1/3	1/3	1/3	1	1	1	2	2	2	0,30	0,873580465	0,076550265
5	1/3	1/3	1/3	1	1	1	2	2	2	0,30	0,873580465	0,076550265
6	1/3	1/3	1/3	1	1	1	2	2	2	0,30	0,873580465	0,076550265
7	1/5	1/5	1/5	1/2	1/2	1/2	1	1	1	0	0,464158883	0,040673397
8	1/5	1/5	1/5	1/2	1/2	1/2	1	1	1	0	0,464158883	0,040673397
9	1/5	1/5	1/5	1/2	1/2	1/2	1	1	1	0	0,464158883	0,040673397

Рисунок 2.6 – Матриця попарних порівнянь альтернатив NNucl

VCPU	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	4	4	4	8	8	8	32768	3,174802104	0,235705716
2	1	1	1	4	4	4	8	8	8	32768	3,174802104	0,235705716
3	1	1	1	4	4	4	8	8	8	32768	3,174802104	0,235705716
4	1/4	1/4	1/4	1	1	1	4	4	4	1	1	0,074242648
5	1/4	1/4	1/4	1	1	1	4	4	4	1	1	0,074242648
6	1/4	1/4	1/4	1	1	1	4	4	4	1	1	0,074242648
7	1/8	1/8	1/8	1/4	1/4	1/4	1	1	1	0	0,314980262	0,023384969
8	1/8	1/8	1/8	1/4	1/4	1/4	1	1	1	0	0,314980262	0,023384969
9	1/8	1/8	1/8	1/4	1/4	1/4	1	1	1	0	0,314980262	0,023384969

Рисунок 2.7 – Матриця попарних порівнянь альтернатив V_{CPU}

Vnet	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	5	5	5	8	8	8	64000	3,419951893	0,250347384
2	1	1	1	5	5	5	8	8	8	64000	3,419951893	0,250347384
3	1	1	1	5	5	5	8	8	8	64000	3,419951893	0,250347384
4	1/5	1/5	1/5	1	1	1	2	2	2	0	0,7368063	0,053935709
5	1/5	1/5	1/5	1	1	1	2	2	2	0	0,7368063	0,053935709
6	1/5	1/5	1/5	1	1	1	2	2	2	0	0,7368063	0,053935709
7	1/8	1/8	1/8	1/2	1/2	1/2	1	1	1	0	0,396850263	0,029050241
8	1/8	1/8	1/8	1/2	1/2	1/2	1	1	1	0	0,396850263	0,029050241
9	1/8	1/8	1/8	1/2	1/2	1/2	1	1	1	0	0,396850263	0,029050241

Рисунок 2.8 – Матриця попарних порівнянь альтернатив Vnet

ΔTfal	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор
1	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
2	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
3	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
4	1	1	1	1	1	1	3	3	3	27,00	1,44224957	0,142857143
5	1	1	1	1	1	1	3	3	3	27,00	1,44224957	0,142857143
6	1	1	1	1	1	1	3	3	3	27,00	1,44224957	0,142857143
7	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048
8	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048
9	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048

Рисунок 2.9 – Матриця попарних порівнянь альтернатив ΔTfal

$Cond_{hw}$	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
2	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
3	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
4	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
5	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
6	1	1	1	1	1	1	3	3	3	27	1,44224957	0,142857143
7	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048
8	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048
9	1/3	1/3	1/3	1/3	1/3	1/3	1	1	1	0	0,480749857	0,047619048

Рисунок 2.10 – Матриця попарних порівнянь альтернатив $Cond_{hw}$

$Serv_{pers}$	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	1	1	1	1	1	1	1	1	0,111111111
2	1	1	1	1	1	1	1	1	1	1	1	0,111111111
3	1	1	1	1	1	1	1	1	1	1	1	0,111111111
4	1	1	1	1	1	1	1	1	1	1	1	0,111111111
5	1	1	1	1	1	1	1	1	1	1	1	0,111111111
6	1	1	1	1	1	1	1	1	1	1	1	0,111111111
7	1	1	1	1	1	1	1	1	1	1	1	0,111111111
8	1	1	1	1	1	1	1	1	1	1	1	0,111111111
9	1	1	1	1	1	1	1	1	1	1	1	0,111111111

Рисунок 2.11 – Матриця попарних порівнянь альтернатив $Serv_{pers}$

N_{vh}	1	2	3	4	5	6	7	8	9	Множення	√ множення	локальний вектор приоритетів
1	1	1	1	1/7	1/7	1/7	1/5	1/5	1/5	2,3324E-05	0,305710709	0,1096694
2	1	1	1	1/7	1/7	1/7	1/5	1/5	1/5	2,3324E-05	0,305710709	0,1096694
3	1	1	1	1/7	1/7	1/7	1/5	1/5	1/5	2,3324E-05	0,305710709	0,1096694
4	7	7	7	1	1	1	1/3	1/3	1/3	12 5/7	1,326352403	0,475810194
5	7	7	7	1	1	1	1/3	1/3	1/3	12 5/7	1,326352403	0,475810194
6	7	7	7	1	1	1	1/3	1/3	1/3	12 5/7	1,326352403	0,475810194
7	5	5	5	3	3	3	1	1	1	3375	2,466212074	0,884718754
8	5	5	5	3	3	3	1	1	1	3375	2,466212074	0,884718754
9	5	5	5	3	3	3	1	1	1	3375	2,466212074	0,884718754

Рисунок 2.12 – Матриця попарних порівнянь альтернатив N_{vh}

C_{rent}	1	2	3	4	5	6	7	8	9	Множення	√ множення	ий вектор приорит
1	1	2	4	5	5	6	6	7	8	403200	4,196001	0,33293
2	0,5	1	2	3	3	3	4	4	5	2160	2,346901	0,186214
3	0,25	0,5	1	1	1	3	3	4	4	18	1,378716	0,109394
4	1/5	0,33333	1	1	2	2	4	4	5	21 1/3	1,40499	0,111478
5	1/5	1/3	1	0,5	1	1	3	3	5	1 1/2	1,046082	0,083001
6	1/6	0,33333	0,33333	0,5	1	1	2	5	5	1/2	0,917991	0,072838
7	1/6	0,25	1	0,5	0,33333	0,5	1	2	3	0	0,650422	0,051608
8	1/7	1/4	0,25	0,25	1/3	0,2	0,5	1	5	0	0,415865	0,032997
9	1/8	0,2	0,25	0,2	0,2	0,2	0,33333	0,2	1	0	0,246281	0,019541

Рисунок 2.13 – Матриця попарних порівнянь альтернатив C_{rent}

Після всіх перетворень відповідно до формул (2.14) – (2.18), отримується вектор ваг альтернатив (табл. 2.5).

Таблиця 2.5 – Вектор ваг альтернатив

№	1	2	3	4	5	6	7	8	9
Вес	0,20	0,199	1,98	0.18	0,094	0.093	0.081	0.081	0.081

Аналіз результатів (табл. 2.5) показує, що найбільш прийнятною в даний момент для початкового завантаження віртуального хоста з вимогами (2.24) є альтернатива 5: Server 2, XP12000 - 2, орендна плата на 30% більше мінімально можливої для цих вимог.

Отже, зазначено, що отримав подальший розвиток метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу». Застосування цього методу базового виділення ресурсів дозволило підвищити рівень балансування завантаження хмарних ресурсів за показником середнього квадратичного відхилення до 8%.

2.5. Висновки за розділом 2

Запропонований підхід до проведення декомпозиції хмарного середовища на зони, виходячи з визначальних особливостей ресурсів, що надаються в кожній зоні. Кожна з зон включає сервери і ресурси зберігання даних зі подібними характеристиками. Розроблена математична модель процесу декомпозиції, що використовує шкалування по окремих типах обчислювальних ресурсів.

Проведено обґрунтування обраного методу для базового виділення хмарних ресурсів – методу аналізу ієрархій. Вибір методу для статичного базового розподілу ресурсів орієнтувався на характерні особливості моделі обслуговування IaaS. Наведені основні переваги та недоліки методу аналізу

ієрархій. Показано, що наведені недоліки методу ієрархій при початковому розподілі ресурсів не є суттєвими. За допомогою обраного методу була здійснена підтримка прийняття рішень у вигляді ієрархічної композиції завдання та рейтингування множини альтернативних рішень.

Запропонований метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу». Запропонований покроковий алгоритм застосування методу аналізу ієрархій, сформована шкала відносин, тобто шкала ступенів значущості параметрів функціонування хмарного середовища. В результаті отримав подальший розвиток метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу», шляхом попередньої декомпозиції множини доступних ресурсів на зони за допомогою введення нерівномірних шкал та використання методу аналізу ієрархій, що дозволяє підвищити рівень балансування завантаження хмарних ресурсів.

Розглянутий приклад базового завантаження віртуального хоста. В результаті застосування запропонованого метода обраний найбільш прийнятний в даний момент для базового завантаження віртуального хоста варіант, що задовольняє вимогам користувача. Отже, запропонований метод дозволив підвищити рівень балансування завантаження обчислювальних ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», за показником середнього квадратичного відхилення до 8%

Основні результати розділу надруковані в наукових працях [2, 4, 6, 10] (додаток А).

.

РОЗДІЛ 3
РОЗРОБКА МЕТОДУ АДАПТИВНОГО РОЗПОДІЛУ ХМАРНИХ РЕСУРСІВ ПРИ ВИКОРИСТАННІ МОДЕЛІ ОБСЛУГОВУВАННЯ «ІНФРАСТРУКТУРА У ЯКОСТІ СЕРВІСУ»

У сучасному світі хмарні обчислювальні ресурси забезпечують різноманітні інформаційні та комунікаційні послуги, а також послуги обробки та зберігання великих обсягів даних. Користувачі можуть отримувати доступ до різноманітних ресурсів та послуг у будь-якому місці та в будь-який час, а також динамічно масштабувати ресурси для оптимізації продуктивності своїх додатків або зниження витрат. Однак попит на хмарні обчислення зростає з кожним роком. У цьому контексті користувачі та розробники все частіше користуються різними характеристиками запитів до хмарних ресурсів. Це стосується різноманітних додатків у реальному часі, які вимагають високої якості та оперативності. Це також стосується обробки критичних обсягів даних, що вимагають підвищеного рівня продуктивності. Крім того, в останні часи стають більш поширеними випадки впливу на стандартні ситуації хмарної обробки різних зловмисних кібердійств. Це призводить до різких непрогнозованих викидів та неочікуваних коливань навантаження на хмарні обчислювальні ресурси.

В цьому контексті ефективний метод розподілу ресурсів може відігравати ключову роль у забезпеченні необхідної якості послуг. Такий метод повинен забезпечувати оперативне обслуговування всіх законних транзакцій та запитів до хмарних ресурсів для гарантування якості обслуговування користувачів, підвищення продуктивності додатків та максимального задоволення вимог замовників.

Крім того, в сучасних умовах актуальним стає також забезпечення безпеки та захисту інформації при розподілі ресурсів. Для цього важливо інтегрувати аспекти кібербезпеки в стратегію прогнозування та запобіжного

розподілу ресурсів, щоб мінімізувати ризики та забезпечити надійний захист даних.

3.1. Узагальнення вимог до методу розподілу хмарних ресурсів при використанні моделі обслуговування «Інфраструктура у якості сервісу»

Процес розподілу ресурсів при використанні технології, що базується на моделі «Інфраструктура у якості сервісу», включає в себе пошук відповідних фізичних серверів для розміщення віртуальних машин (ВМ). У попередніх дослідженнях часто використовувалися базові евристичні алгоритми, такі як round robin (RR) [51], best fit (BF) [43] і min-max [59] для вирішення проблеми розподілу хмарних ресурсів на невеликих платформах. Проте такі методи часто неефективно використовують ресурси, особливо в разі великих хмарних інфраструктур, які зазвичай використовуються при застосуванні даної технології.

В умовах невизначеності та ризиків зовнішніх кібервпливів стає важливим враховувати можливі загрози при прогнозуванні та розподілі ресурсів. Одним із класичних методів, які застосовуються для вирішення такої складної задачі розподілу хмарних ресурсів, є так звана "задача упаковки баків" (bin packing problem). Вона полягає у оптимальному упакуванні 'n' об'єктів в 'm' контейнерів з мінімальним використанням контейнерів. Також пропонується динамічний метод упаковки баків, який дозволяє знизити витрати на хмарні ресурси шляхом закриття порожніх контейнерів [94].

Іншим перспективним підходом до вирішення задачі ефективного розподілу хмарних ресурсів є моделювання процесу розміщення віртуальних машин (ВМ) як математичної багатокритерійної задачі оптимізації [42]. Цей метод спрямований на знаходження оптимального рішення, з урахуванням одночасно декількох різних критеріїв та обмежень.

Основна ідея полягає в тому, щоб виразити складну задачу розподілу хмарних ресурсів у вигляді математичної функції, яка враховує кілька

важливих параметрів та показників. Ці параметри можуть включати такі характеристики, як завантаження серверів, використання центральних процесорів (CPU), обсяг доступної пам'яті, розміри дискового простору та інші. Важливо, щоб ця функція відображала та балансувала різноманітні потреби додатків та користувачів, а також оптимізувала використання наявних ресурсів [42].

Такий підхід забезпечує більш гнучке та адаптивне рішення задачі розподілу ресурсів, здатне враховувати різноманітні вимоги додатків, зміни навантаження та можливі зовнішні впливи, включаючи кіберзагрози. Проте вибір та налаштування багатокритеріальних функцій та параметрів алгоритму потребує ретельної роботи та експертних знань, що ускладнює практичну реалізацію цього методу.

Науковий аналіз використаних джерел за темою розкриває різноманітні підходи та тенденції в області оптимізації розподілу ресурсів у хмарних обчисленнях. Дослідження в цій галузі спрямовані на поліпшення продуктивності, вартості, адаптивності та безпеки хмарних середовищ.

У роботі [51] досліджується використання методів динамічного розподілу хмарних ресурсів. Вони показують, що зміна розподілу ресурсів під час виконання завдань може сприяти досягненню кращих результатів та зменшенню витрат. В той же час робота має аналітичний характер, та не пропонує нових технічних рішень, для виконання завдань оптимізації розподілу хмарних ресурсів.

Робота [43] присвячена дослідженню різних методів динамічного розподілу ресурсів в хмарних обчисленнях. Вона визначає переваги та недоліки різних підходів, таких як механізми прийняття рішень, алгоритми розподілу та прогнозування вимог до ресурсів. Однак, як і в попередній статті, автори лише розглядають майбутні можливості методів динамічного розподілу хмарних ресурсів, без врахування практичного досвіду та можливих ризиків, наприклад ризиків кібербезпеки.

Відповідно до дослідження [59], розподіл ресурсів може бути налаштований автоматично в залежності від змінного завантаження хмарних обчислювальних вузлів. Цей підхід дозволяє забезпечити оптимальний розподіл ресурсів та уникнути перевантажень чи недостатнього використання ресурсів. Недоліком запропонованого авторами метода є нехтування питаннями безпеки та відсутність врахування ризиків зловмисних кібервторгнень.

Дослідження [94] показують, що адаптивний розподіл ресурсів може бути важливим для оптимізації обробки великих обсягів даних. Для цього пропонується використовувати алгоритми, які аналізують завантаження та динамічно розподіляють ресурси залежно від об'єму даних та обчислювальних завдань. Незважаючи на перспективність запропонованого напрямку досліджень, стаття передбачає оптимізацію лише певних хмарних ресурсів (ЦП та пам'яті). В роботі не враховуються інші, так само важливі ресурси, такі як пропускна здатність мережі або ресурси фаерволу.

В роботі [42] вивчається проблема балансування навантаження в хмарних обчисленнях. Представлено огляд різних алгоритмів балансування навантаження та аналізує їхні переваги та недоліки. Однак запропоновані методи оптимізації можуть ґрунтуватися на припущенні, що всі хмарні додатки та навантаження є схожими, і не враховувати різноманітності вимог до ресурсів та поведінки виконання різних додатків. Ця спрощена модель може обмежувати застосовність методів до гетерогенних навантажень.

В роботі [7] формалізується модель розподілу хмарних ресурсів, що охоплює різні аспекти оптимізації ресурсів у хмарних обчисленнях, включаючи алгоритми прийняття рішень, оптимізаційні методи та ефективність використання ресурсів. В той же час масштабованість запропонованих методів оптимізації не досліджувалась. Хоча автори статті наводять обіцяючі результати в експериментах меншого масштабу, можливо, не було повністю розглянуто, наскільки добре ці методи можна застосовувати та масштабувати для обробки більших та більш вимогливих хмарних інфраструктур.

В статті [92] розглядаються проблеми та виклики управління ресурсами в хмарних середовищах. Авторами запропоновано аналіз різних підходів до адаптивного розподілу ресурсів та обговорює можливі шляхи вирішення викликів. Однак автори обмежують свою область дослідження тільки адаптивним управлінням ресурсами для "spot workers" (робітників на спотовому ринку) в хмарних середовищах, залишаючи осторонь інші аспекти ресурсного управління та хмарних сервісів. Крім того, в статті не розглядається широкий спектр алгоритмів і технік адаптивного управління ресурсами, що може обмежити комплексність та точність результатів.

В доповіді [81] описується метод адаптивного управління ресурсами в хмарних середовищах.

В доповіді розглядається використання прогнозування, оптимізаційних алгоритмів та автоматичного контролю для досягнення оптимального використання ресурсів. В той же час автори доповіді не враховують зовнішні кібервтручання, як один з основних збуджуючих та негативних факторів, що впливає на зміни в середовищі, та має вплив на ефективність запропонованого методу.

Додатково, дослідження в області математичного моделювання процесів в хмарних середовищах автори статті [107] демонструють, що впровадження систем паралельного обчислення може значно спростити та оптимізувати розподіл ресурсів, зменшити витрати на обслуговування та забезпечити ефективне використання обчислювальних ресурсів. Однак, як і попередні дослідники автори нехтують питаннями можливих ризиків кібервтручання. Це знижує практичну цінність результатів дослідження.

Ще одна стаття [86] присвячена сучасному використанню методу оптимізації на основі технології «туманних» обчислень. Ця технологія допомагає підвищити точність результатів моделювання.

В той же час, використання цієї технології у сукупності з методами прогнозування в рамках розподілу хмарних ресурсів авторами не розглядалась.

Доповідь [80] присвячена розробці моделі динамічного управління телекомунікаційними та комп'ютерними ресурсами. Доповідь надає комплексний погляд на динамічне управління телекомунікаційними та комп'ютерними ресурсами, що може допомогти розкрити більше аспектів цієї проблематики. Присутність моделі управління забезпечує систематичний підхід до розгляду ситуацій, пов'язаних з ресурсами, та розробку оптимальних стратегій.

Основний акцент на динамічному управлінні вказує на важливість адаптації до змінних умов та потреб. В той же час, доповідь не надає достатньо деталей щодо конкретних аспектів моделі та її реалізації. Доповідь не містить практичних випробувань або застосування моделі на реальних даних, це обмежує її значущість.

В цілому можна відзначити, що в умовах динамічних та активних запитів більшість хмарних систем зіштовхуються з труднощами у забезпеченні швидкого розподілу ресурсів та зниження втрат ресурсів серверів [21, 34, 76].

Тому удосконалений метод розподілу хмарних ресурсів повинен задовольняти наступним вимогам для динамічних запитів до ресурсів [63, 70]:

1. Швидка обробка багатьох динамічних запитів до ресурсів, включаючи реагування на зловмисні кібервпливи.
2. Надання оптимальних ресурсів для задоволення запитів.
3. Мінімізація використання фізичних серверів та рівномірний розподіл різних типів ресурсів (ядра ЦП, обсяг пам'яті та розмір диска) для зниження втрат ресурсів.

3.2. Структура методу адаптивного розподілу хмарних ресурсів

На рис. 3.1 зображено послідовність виконання основних етапів запропонованого методу адаптивного розподілу хмарних ресурсів.

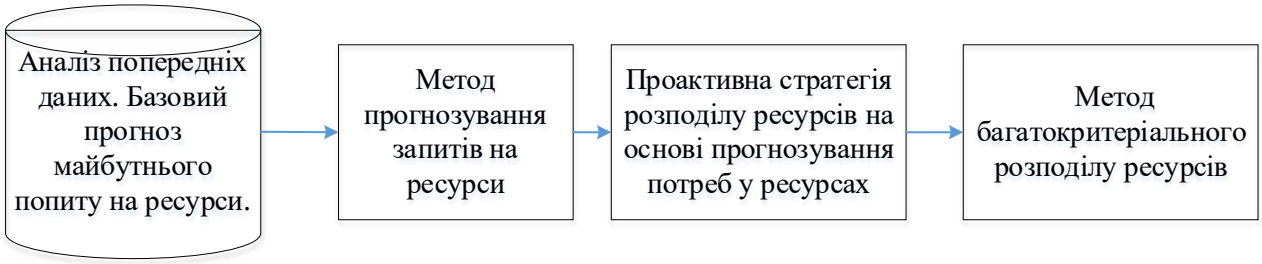


Рисунок 3.1 – Послідовність виконання основних етапів запропонованого методу адаптивного розподілу хмарних ресурсів

У цій схемі реалізується метод прогнозування запитів на ресурси на основі тесту на послідовність серій, стратегія запобіжного розподілу ресурсів на основі прогнозу запитів на ресурси і формалізується метод багатокритеріального розподілу ресурсів, який вирішується з використанням вдосконаленого алгоритму NSGA-II (Nondominated Sorting Genetic Algorithm II) [118].

На першому етапі тест на послідовність серій використовується для виявлення різноманітних варіацій та закономірностей в послідовності запитів на ресурси, а також можливих аномалій, що можуть свідчити про потенційні кібератаки. Це дозволяє встановити залежності між запитами та виявити незвичайні зміни в майбутніх запитах, що може бути зв'язано зі зловмисними діями [69]. На основі цього аналізу будується прогноз, який не лише служить основою для розподілу ресурсів, але й допомагає виявити та запобігти можливим кіберпосяганням.

Таким чином, враховуючи можливість зловмисних кібервтручань [47], цей метод спрямований не лише на передбачення майбутніх ресурсних запитів, а й на ідентифікацію потенційних загроз та реалізацію відповідних заходів безпеки.

Другий етап методу адаптивного розподілу ресурсів передбачає реалізацію стратегії превентивного розподілу ресурсів на основі прогнозування запитів на ресурси. Для досягнення оптимального використання ресурсів та

забезпечення ефективної реалізації завдань системи можна використовувати такі методи та підходи:

– Аналіз попередніх даних: Використання попередніх даних запитів на ресурси дозволяє виявити патерни та тенденції в використанні ресурсів. Це допомагає створити базовий прогноз майбутнього попиту на ресурси.

– Часові ряди та статистичні методи: Використання методів аналізу часових рядів, таких як ARIMA (AutoRegressive Integrated Moving Average) [75], може допомогти прогнозувати майбутні потреби в ресурсах на основі змін у часі та зв'язків між даними.

– Машинне навчання: Використання алгоритмів машинного навчання, таких як нейронні мережі, може допомогти покращити точність прогнозів шляхом виявлення складних залежностей у даних.

– Методи оптимізації: Використання методів оптимізації, таких як генетичні алгоритми або методи лінійного програмування, може допомогти знайти оптимальний розподіл ресурсів з урахуванням різноманітних обмежень та критеріїв.

– Врахування динамічності: Прогнозування може бути покращено шляхом врахування динамічних змін у використанні ресурсів, таких як відповідь на навантаження чи пікові навантаження.

– Реакція на кризові ситуації: Передбачені резерви ресурсів можуть бути виділені для реагування на несподівані навантаження або кризові ситуації.

Але треба враховувати, що основним критерієм при реалізації стратегії превентивного розподілу ресурсів на основі прогнозування запитів на ресурси є час виконання відповідного застосунку, котрий використовує результати короткочасного прогнозу. Виходячи з цього в роботі пропонується використання першого підходу дослідження (аналіз попередніх даних).

Третій етап методу адаптивного розподілу ресурсів передбачає вибір, вдосконалення та формалізацію алгоритму оптимізації. Для вирішення багатокритеріальної задачі розподілу ресурсів використовується удосконалений алгоритм NSGA-II.

Цей алгоритм використовує ідеї генетичних алгоритмів та сортування за недомінованими рішеннями для знаходження наближених до оптимальних рішень за рядом критеріїв одночасно. Однією з важливих частин цього етапу є підбір оптимальних параметрів алгоритму. Параметри алгоритму впливають на його продуктивність та здатність знаходити ефективні рішення.

3.3. Метод адаптивного прогнозування на основі тесту на послідовність серій

Метод адаптивного прогнозування має на меті вирішення таких завдань як зменшення часу прогнозування та покращення точності прогнозу. Порядок виконання кроків процедури прогнозування представлений на рис. 3.2.

З послідовності запитів на віртуальні машини (ВМ) вилучаються *m* компонентних послідовностей за допомогою методу аналізу головних компонент. Даний метод був використаний для визначення взаємозв'язків між досліджуваними послідовностями та виявлення прихованих факторів, що сприяють кореляційним залежностям та статистичним зв'язкам між ними, що дозволило сформувати необхідні компонентні послідовності з використанням програми «Statistika». Далі ці компонентні послідовності виявляються для пошуку та попередньої обробки збурень.

На наступному кроці розраховуються значення адаптивного передбачення для цих попередньо оброблених послідовностей. На прикінці, ці послідовності прогнозуються шляхом адаптивного вибору методу EEMD-ARIMA або EEMD-RT-ARIMA [75, 91] залежно від порівняння їх значень адаптивного прогнозування з наперед встановленими порогами.

EEMD-ARIMA (Ensemble Empirical Mode Decomposition - AutoRegressive Integrated Moving Average) – це метод, який використовується для аналізу та передбачення часових рядів. Даний метод об'єднує дві техніки: ансамблеву емпіричну декомпозицію ряду (EEMD) та авторегресійну інтегровану рухому середню (ARIMA).

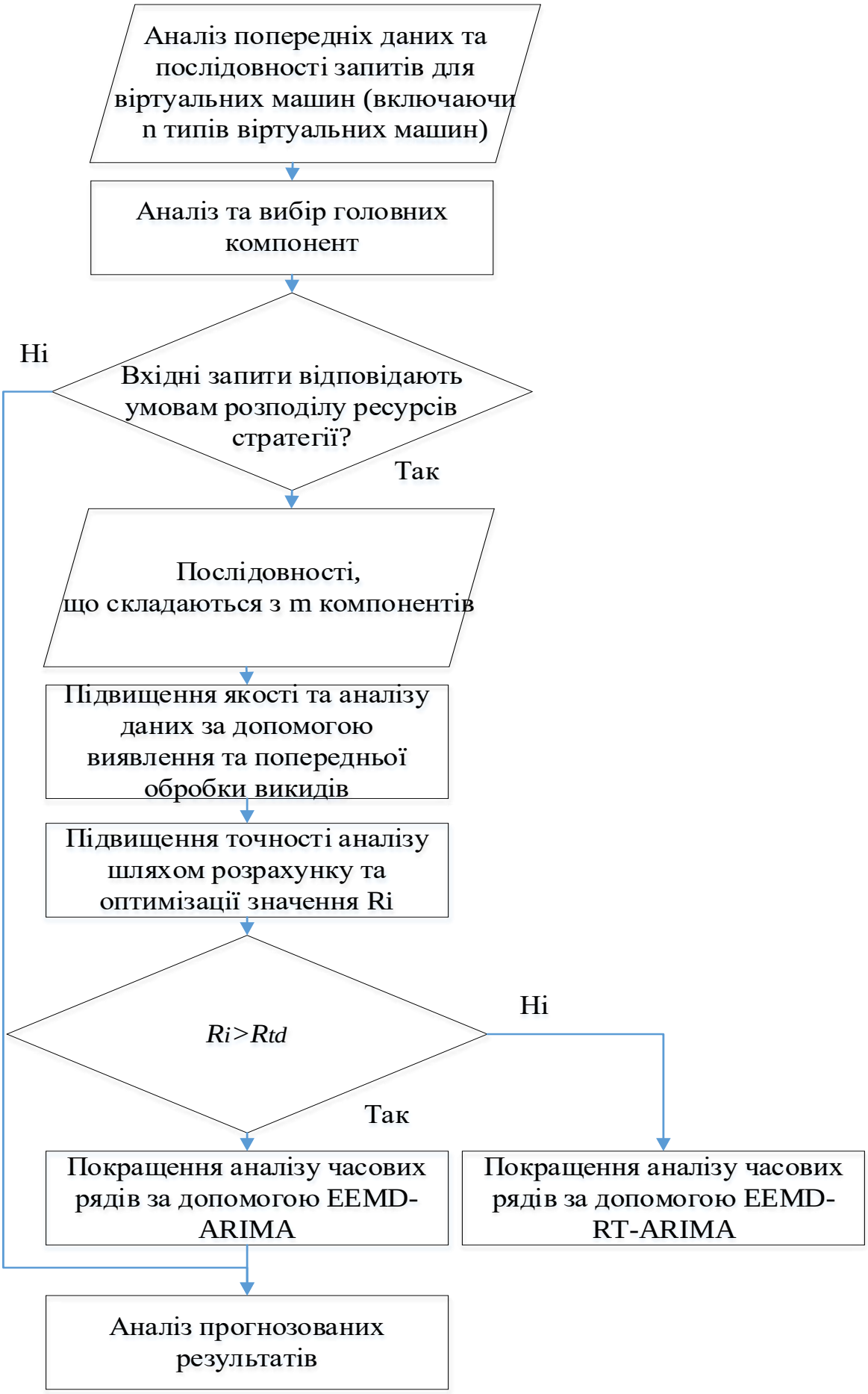


Рисунок. 3.2 – Порядок виконання кроків процедури прогнозування

EEMD (Ensemble Empirical Mode Decomposition) – це метод, який допомагає розкласти складний часовий ряд на кілька простіших складових частин, відомих як "емпіричні моди". Цей процес дозволяє виділити локальні варіації в ряді та зробити його аналіз більш зрозумілим.

ARIMA (AutoRegressive Integrated Moving Average) – це статистичний метод аналізу та передбачення часових рядів. Він поєднує авторегресійну (AR) компоненту, інтегровану (I) компоненту та рухому середнього (MA) компоненту для моделювання та передбачення залежностей в часовому ряді.

Етапи використання EEMD-ARIMA включають:

– Застосування EEMD для декомпозиції часового ряду на емпіричні моди.

– Вибір підходящих емпіричних мод для подальшого аналізу та передбачення.

– Застосування ARIMA до обраного емпіричного моду для передбачення майбутніх значень ряду.

В свою чергу в метод EEMD-RT-ARIMA (Ensemble Empirical Mode Decomposition - Running Trends - AutoRegressive Integrated Moving Average) крім зазначених процедур доєднається ще – RT (Running Trends). Це набір процедур, який визначає тенденції або напрямки руху в часовому ряді. Використовуючи цей набір можна виявити основні зміни та тенденції в ряді.

В методі EEMD-RT-ARIMA спочатку використовується EEMD для декомпозиції часового ряду на емпіричні моди, після чого визначаються тенденції за допомогою RT. Далі застосовується ARIMA до емпіричних мод та їхніх тенденцій для передбачення майбутніх значень ряду.

Хмарна платформа надає багато варіантів віртуальних машин, таких як 8CPU16G (8 ядер CPU, 16 ГБ пам'яті) та 16CPU32G (16 ядер CPU, 32 ГБ пам'яті) [49, 50], прогнозування кожного конкретного типу запитів на віртуальній машині стає неможливим через тривалий час прогнозування. Отже, в методі прогнозування на першому етапі потрібно зробити аналіз головних компонент. Цей аналіз дозволяє виділити основні послідовності компонент та

ефективно використати час та підвищити оперативність процесу прогнозування.

Для прикладу, розглянемо послідовність запитів на віртуальній машині з n типами віртуальних машин, позначену як

$$S=(s_1, ..., s_i, ..., s_k), \quad (3.1)$$

де s_i представляє кількість віртуальних машин у і-му запиті.

З цієї послідовності S можна виділити послідовність компонент

$$S_h=(s_{h1}, ..., s_{hi}, ..., s_{hk}) \quad (3.2)$$

для типу віртуальних машин h, де s_{hi} позначає кількість віртуальних машин даного типу, котрі приймають участь у реалізації i-го запиту.

Для реалізації прогнозування запитів на віртуальній машині обираються мінімальні послідовності компонент. Загальна кількість запитів у цих послідовностях, виражена як частка запитів. Вона перевищує заздалегідь визначений поріг T_{td} на кожній точці вибірки. Обрані послідовності компонент розглядаються як основні компоненти.

Припустимо, що у нас є дві послідовності компонент:

$$S_d=(s_{d1}, ..., s_{di},...,s_{dk}) \text{ і } S_g=(s_{g1}, ..., s_{gi}, ..., s_{gk}). \quad (3.3)$$

В цьому прикладі s_{di} і s_{gi} представляють кількість різних типів запитів на виртуальній машині.

Для $\forall s_{di} \in S_d$ та $s_{gi} \in S_g$, якщо співвідношення $(s_{di}+s_{gi})/s_i$ більше або дорівнює передвстановленому порогу T_{td}, то відбувається вибіркове об'єднання компонентних послідовностей S_d та S_g в основну послідовність S_{head}.

Ці дві компонентні послідовності обираються та об'єднуються у множину S_{head}, включаючи ключові компонентні послідовності для виконання прогнозу.

Важливо зауважити, що підвищення значення порогового показника T_{td} призводить до відбору більшої кількості компонентних послідовностей. Це з свого боку забезпечує більш точний прогноз запитів на віртуальні машини та, відповідно, дозволяє досягти більш точного розподілу ресурсів. Але зростання числа компонентних послідовностей призводить до підвищення вартості прогнозування [121].

Для ілюстрації, якщо поріг встановлено як T'_{td} , що перевищує вихідне порогове значення T_{td} :

$$\forall s_{di} \in S_d \text{ , } s_{gi} \in S_g \text{ та } s_{hi} \in S_h, (s_{di} + s_{gi} + s_{hi})/s_i > T'_{td}, \quad (3.4)$$

то до набору S_{head} можуть включити три ключові компонентні послідовності – S_d, S_g та S_h. Цей процес відбору виконується для кожного значення з цих компонентних послідовностей.

Потім кожну компонентну послідовність розбивають на декілька підпослідовностей для проведення прогнозування з використанням методів EEMD-ARIMA або EEMD-RT-ARIMA.

Вибір між EEMD-ARIMA та EEMD-RT-ARIMA здійснюється в залежності від порівняння значень часу виконання R_i (тест на послідовність серій) та порогового значення R_{td}. Якщо значення R_i для конкретної компонентної послідовності перевищує порогове значення, то використовується метод EEMD-ARIMA.

В іншому випадку, якщо значення R_i знаходиться нижче порогового значення, то застосовується метод EEMD-RT-ARIMA.

Це дозволяє адаптивно вибирати найбільш відповідний метод прогнозування залежно від характеру даних та їхньої змінюваності, з урахуванням особливостей запитів до ресурсів.

Припустимо, що компонентну послідовність розбито на m підпослідовностей, і кожна підпослідовність потребує n секунд для завершення прогнозування. Оскільки час прогнозування для кожної підпослідовності

практично однаковий, ми припускаємо, що для кожної підпослідовності потрібно n секунд.

Якщо дві компонентні послідовності, S_d та S_g, вибрані на основі порогового значення T_{td}, вартість прогнозування може бути розрахована наступним чином.

$$C(S_d, S_g, T_{td}) = 2m\times. \tag{3.5}$$

Однак, якщо три компонентні послідовності S_d, S_g та S_h вибрані нижче порогового значення T'_{td}, то вартість прогнозування буде розрахована за допомогою наступного виразу.

$$C'(S_d, S_g, S_h, T'_{td}) = 3m\times n. \tag{3.6}$$

Проведені дослідження вказують на те, що збільшення порогу T_{td} та, відповідно, розширення множини компонентних послідовностей може підвищити точність прогнозування. Однак це значно збільшить вартість прогнозу та призведе до збільшення часових затрат на процедури розподілу ресурсів.

Одним з негативних наслідків таких дій може стати перехід з нормального режиму роботи додатків до аномального режиму, а це, в свою чергу, підвищить ризик зовнішніх кіберзагроз. Тому встановлення порогу T_{td} має важливе значення, включаючи аспекти кібербезпеки.

Можна припустити, що для прогнозування майбутніх запитів на віртуальні машини було вибрано p основних компонентних послідовностей. Необрана компонентна послідовність S_h має більше запитів на віртуальні машини, ніж інші компонентні послідовності. Поріг T_{td} може бути наближений до мінімального значення відношення запитів на віртуальні машини згідно з наступною формулою за певних умов.

Таким чином, вибір порогу T_{td} впливає як на точність результатів прогнозу, так і на оперативність операцій прогнозування. При цьому оптимізаційне завдання мінімізації T_{td} можно формалізувати наступним чином

$$T_{td} \leftarrow min((s_{11} + ... + s_{1k})/s_1, ..., (s_{i1} + ... + s_{ik})/s_i...,(s_{z1} + ... + s_{zk})/s_z), \quad (3.7)$$

при обмеженнях:

$$s_{hi}/s_i < \varepsilon_h, \quad (3.8)$$

$$1/(z+1) > \varepsilon_t. \quad (3.9)$$

У цій формалізації оптимізаційної задачі параметри ε_h та ε_t вказують на порогові значення відношення запитів на віртуальні машини та відношення витрат часу на прогнозування, відповідно. Якщо додаткова доля запитів на віртуальні машини s_{hi}/s_i перевищує порогове значення ε_h та додаткове відношення витрат часу на прогнозування $1/(z+1)$ менше порогового значення ε_t, то компонентна послідовність S_h буде вибрана для прогнозування майбутніх запитів на віртуальні машини.

Для виявлення динамічних сплесків у цих основних компонентних послідовностях пропонується використовувати метод квартилів. На першому етапі процедури розраховуються перший квартиль (*Q1*), третій квартиль (*Q3*) і міжквартильний розмах (*IQR*) за формулою $IQR = Q3 - Q1$.

На другому етапі визначаються динамічні сплески, які перевищують значення *Q3* в 1,5 рази або знаходяться нижче *Q1* в 1,5 рази. На третьому етапі проводиться заміна виявлених динамічних сплесків. Це дало змогу покращити обробку даних і достовірність прогнозу. Цей процес здійснюється за допомогою методу інтерполяції кубічним сплайном, котрий обраний як найбільш прийнятний варіант між складністю та точністю розрахунків. Інтерполяція кубічним сплайном - це математичний метод, що дозволяє

апроксимувати пропущені або викинуті значення у даних, створюючи плавний і неперервний перехід між відомими точками.

На прикладі розробленого методу, інтерполяція кубічним сплайном застосовується для створення нових значень, які найкраще відповідають вже наявним точкам даних.

Це допомагає видалити недостовірні сплески та згладити часові ряди, що є важливим для більш точного прогнозування майбутніх запитів на віртуальні машини [123, 124].

Попередньо оброблені компонентні послідовності аналізуються за допомогою методу RT. На наступному етапі реалізується адаптивний метод прогнозування на основі RT (APMRT). Якщо значення R_i компонентної послідовності перевищує заздалегідь встановлений поріг (R_{td}), для прогнозування майбутніх запитів на ресурси вибирається метод EEMD-ARIMA. В іншому випадку, якщо значення R_i менше порога, для прогнозування використовується метод EEMD-RT-ARIMA. Такий підхід дозволяє підвищити точність прогнозування шляхом попередньої обробки динамічних сплесків у основних компонентних послідовностях та вибору більш точного методу прогнозування.

Точне прогнозування майбутньої кількості кожного типу запитів на віртуальні машини дозволяє проводити превентивний розподіл ресурсів, забезпечуючи своєчасність та ефективність розподілу ресурсів.

Оцінимо часові витрати при практичному застосуванні розробленого методу.

В даному методі часова складність видобутку компонентної послідовності становить $O(k)$. Отже, часова складність видобутку m компонентних послідовностей та попередньої обробки даних становить $O(m \times k)$.

Визначимо значення R_i для видобутих послідовностей та виконаємо прогноз з використанням адаптивного алгоритму прогнозування APMRT. Часові витрати стають

$$O(2m \times k + q \times m \times k) \quad (3.10)$$

або

$$O(2m \times k + z \times m \times k), \quad (3.11)$$

де q і z окремо позначають кількість розкладених компонентних послідовностей та нових відновлених компонентних послідовностей ($z < q$).

Отже, часові витрати алгоритму APMRT становлять $O(Q \times m \times k)$ або $O(P \times m \times k)$, що менше часової складності $O(Q \times n \times k)$ або $O(P \times n \times k)$ для всіх n компонентних послідовностей. Такий підхід дозволяє зменшити час прогнозування шляхом видобутку основних компонентних послідовностей до 1,5 разів.

3.4. Метод превентивного формування черг запитів на віртуальні машини хмарного середовища

Стратегія активного розподілу хмарних ресурсів спрямована на оперативне прогнозування майбутніх запитів на ресурси та своєчасне адаптування процедур розподілу до динамічних сплесків (змін) запитів у майбутньому [125]. Це дозволить ефективно протидіяти непередбачуваним або аномальним ситуаціям, включаючи кібернебезпечні інциденти. Особливо важливе місце у цій стратегії займає метод превентивного формування черг запитів на віртуальні машини, узагальнена структурна схема якого наглядно представлена на рис. 3.3.

Суть цього методу полягає у застосуванні процедур адаптивного прогнозування на основі аналізу попередніх даних і базується на параметрі R_i (час відповіді). Основна увага методу спрямована на формування гібридної черги запитів на віртуальні машини. При цьому така черга формується з урахуванням актуальних запитів, а також передбачених майбутніх динамічних змін. Зазвичай, черга формується на основі короткочасного прогнозування на часовому інтервалі, що містить порядку 100 відліків.

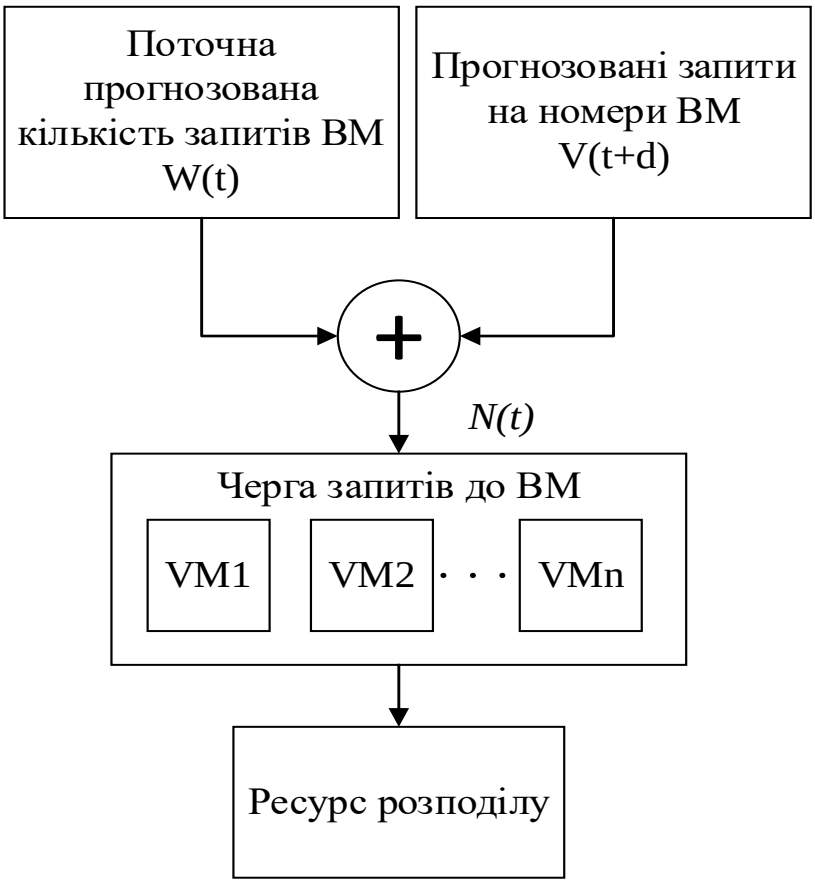


Рисунок 3.3 – Структурна схема методу
превентивного формування черг запитів на віртуальні машини

Припустимо, що поточна послідовність запитів на віртуальні машини позначається як

$$Rec(t)=(rec_1(t),...,rec_i(t),...rec_n(t)), \quad (3.12)$$

де $rec_i(t)$ представляє кількість віртуальних машин типу i в момент часу t.

Для оцінки майбутньої кількості h основних типів запитів на віртуальні машини на момент часу $t + d$ використовується адаптивний алгоритм прогнозування APMRT, який дає наступне позначення: $V_i(t + d)$ представляє кількість запитів i-го основного типу на віртуальні машини на момент часу $t + h$. Загальну кількість запитів на віртуальні машини $N(t)$ в момент часу t можна формалізувати як суму поточної кількості запитів $Rec(t)$ та передбаченої кількості запитів $V(t+d)$:

$$V(t+d) = V_1(t+d) + ... + V_i(t+d) + ... + V_h(t+d), \quad (3.13)$$

де $V_i(t+d)$ – *і-й* основний тип запитів на віртуальні машини на час $t+d$. Загальна кількість запитів на віртуальні машини $N(t)$ в час t повинна дорівнювати сумі поточної кількості запитів на віртуальні машини $Rec(t)$ та передбаченої кількості запитів на віртуальні машини $V(t + d)$:

$$N(t) = W(t) + V(t + d)\times C(t)\times P(t), \quad (3.14)$$

де $W(t)= rec_1(t) + ... + rec_i(t) + ... + rec_n(t)$ – поточна кількість запитів на віртуальні машини в час t.

Якщо передбачена кількість запитів на віртуальні машини $V(t+d)$ не менше порога N_{td}, деякій віртуальні машини повинні бути виділені ресурси заздалегідь. Параметр $C(t)$ повинен дорівнювати 1, а $P(t)$ – це відсоток (наприклад, 25%) від запитів на віртуальні машини, які мають бути виділені ресурси заздалегідь з урахуванням передбаченої кількості запитів на віртуальні машини $V(t+d)$. В іншому випадку, не потрібно надавати віртуальні машини заздалегідь, тобто $C(t) = 0$. Після визначення передбаченої кількості запитів на віртуальні машини $V(t+d)$, слід встановити послідовність запитів на віртуальні машини.

Припустимо, що передбачена кількість запитів на віртуальні машини впорядкована за спаданням від першого типу віртуальної машини 1 до h. Найбільші запити на віртуальні машини (тобто запити першого типу) розташовуються на початку послідовності запитів, а найменші запити (тобто запити h типу) розташовуються в кінці послідовності запитів на віртуальні машини. Передбачена послідовність запитів на віртуальні машини може бути виражена наступним чином:

$$Rec(t+d)= \begin{pmatrix} rec_1^1(t+d), rec_2^1(t+d), \ldots, rec_j^i(t+d), \\ rec_{j+1}^i(t+d), \ldots, rec_m^h(t+d) \end{pmatrix}, \quad (3.15)$$

де $rec_j^i(t+d)$ – це кількість запитів *j*-го типу віртуальних машин. Таким чином, послідовність запитів на віртуальні машини в момент часу *t* може бути виражена наступним чином:

$$Rec'(t+d)= = \begin{pmatrix} rec_1(t), \ldots, rec_n(t), rec_1^1(t+d), rec_2^1(t+d), \ldots, \\ rec_j^i(t+d), rec_{j+1}^i(t+d), \ldots, rec_m^h(t+d) \end{pmatrix}, \quad (3.16)$$

Отже, метод превентивного формування черг запитів на віртуальні машини хмарного середовища можна розбити на декілька етапів.

Етап 1: Прогнозування майбутніх запитів на віртуальні машини.

– Використовуючи адаптивний метод прогнозування на основі аналізу попередніх даних та параметра R_i, визначення передбаченої кількості майбутніх основних типів запитів на віртуальні машини на певний час в майбутньому *t+d*.

– Позначення кількості запитів $V_i(t+d)$ *i*-го основного типу віртуальних машин в момент часу *t+d*.

– Обчислення загальної кількості запитів на віртуальні машини *N(t)* в момент часу *t* шляхом додавання поточної кількості запитів *Res(t)* і передбаченої кількості запитів *V(t+d)*.

Етап 2: Визначення необхідності заздалегідь виділяти ресурси.

– Розрахунок поточної кількості запитів на віртуальні машини *W(t)* в момент часу *t*.

– Якщо передбачена кількість запитів на віртуальні машини $V(t+d)$ не менше порогового значення N_{td}, видача ресурсів заздалегідь для деяких віртуальних машин.

– Встановлення параметра $C(t)$ та параметра $P(t)$, які слід виділити заздалегідь відносно передбаченої кількості $V(t+d)$.

Етап 3: Встановлення послідовності запитів на віртуальні машини.

– Врахування передбаченої кількості запитів на віртуальні машини $V(t+d)$ та встановлення послідовності запитів.

– Припущення, що передбачена кількість запитів на віртуальні машини впорядкована за спаданням від першого типу віртуальної машини до h, зіставлення найбільших запитів на початку послідовності, а найменших запитів в кінці.

– Знаходження передбаченої послідовності запитів на віртуальні машини та послідовності запитів на віртуальні машини в момент часу t.

3.5. Модель багатоцільового розподілу ресурсів

Модель багатоцільового розподілу ресурсів є математично і структурно формалізованим набором алгоритмів та процедур, що акцентують на плануванні завдань та балансуванні навантаження ресурсів [9].

Модель складається з трьох основних частин, призначених для формалізації завдань.

У першій частині створюється стекова таблиця, що містить інформацію про всі хмарні запити та час їх виконання на доступних віртуальних машинах, як показано на рис. 3.4.

У другій частині вирішується завдання мінімізації ресурсів для обслуговування всіх запитів за допомогою трьох розглянутих методів планування: min-min, max-min і генетичного, як показано на рис. 3.5, 3.6 і 3.7 відповідно.

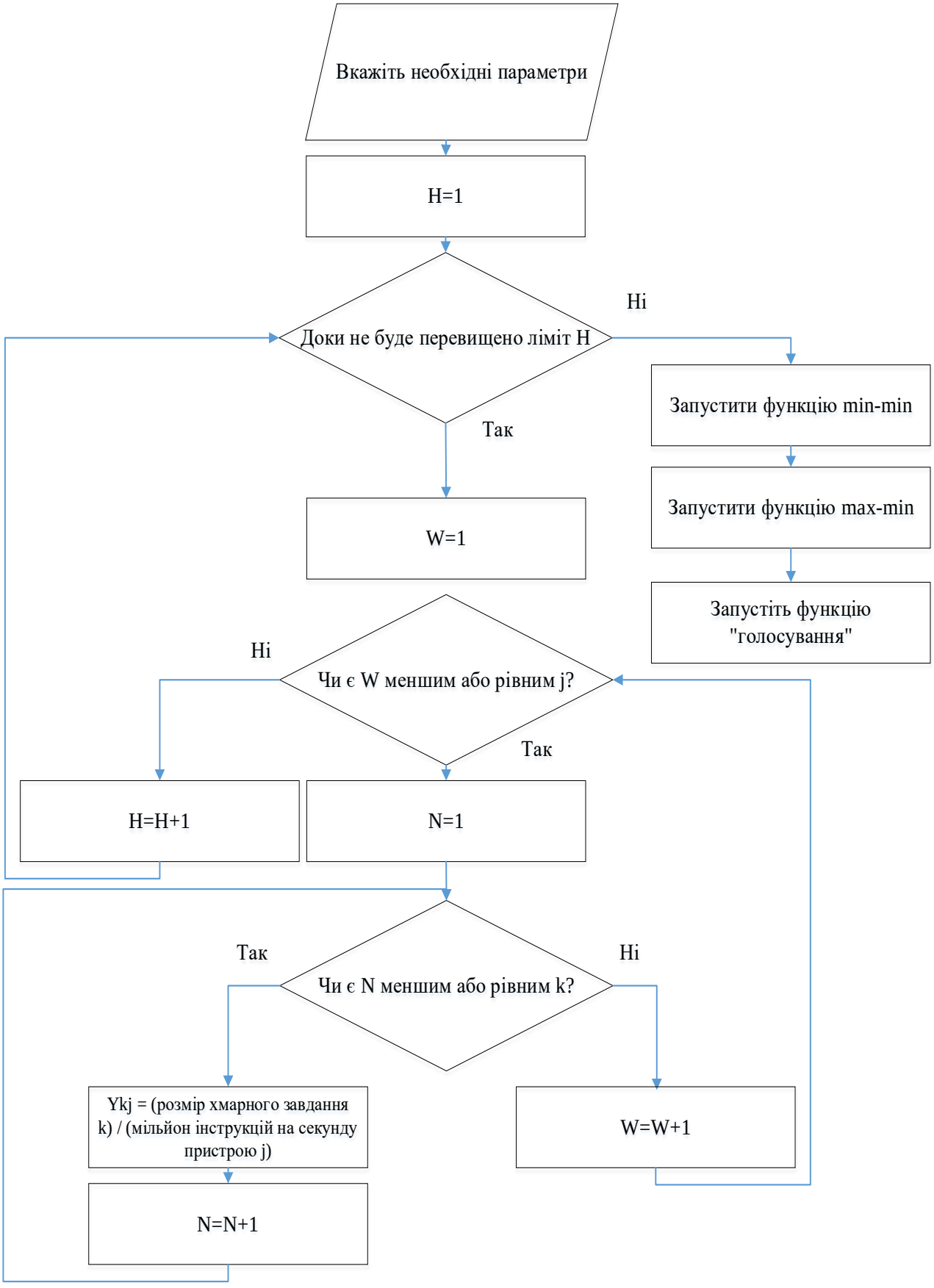


Рисунок 3.4 – Порядок виконання кроків формування стекової таблиці

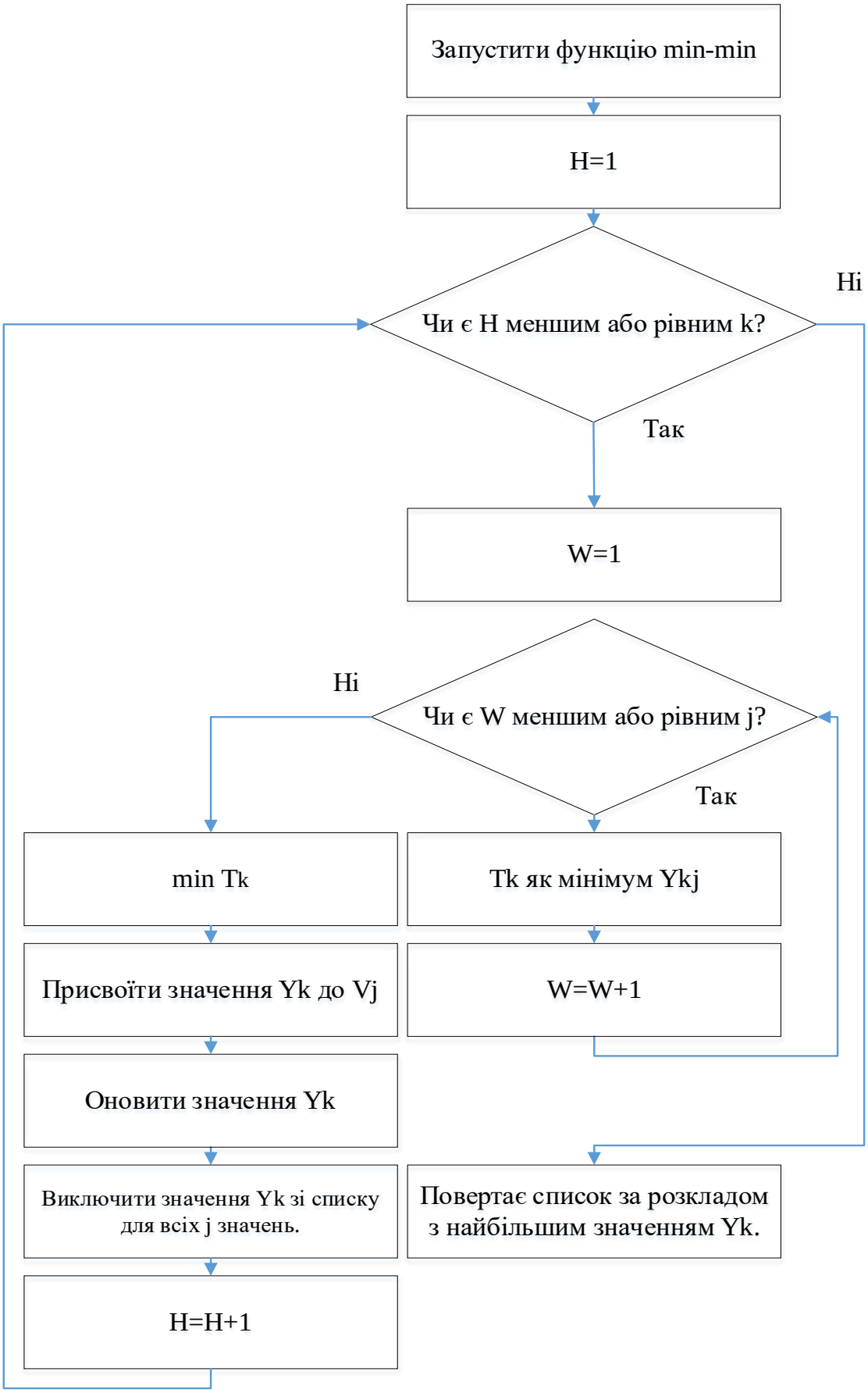


Рисунок 3.5 – Порядок виконання кроків процедури мінімізації ресурсів для обслуговування всіх запитів за допомогою методу «min-min»

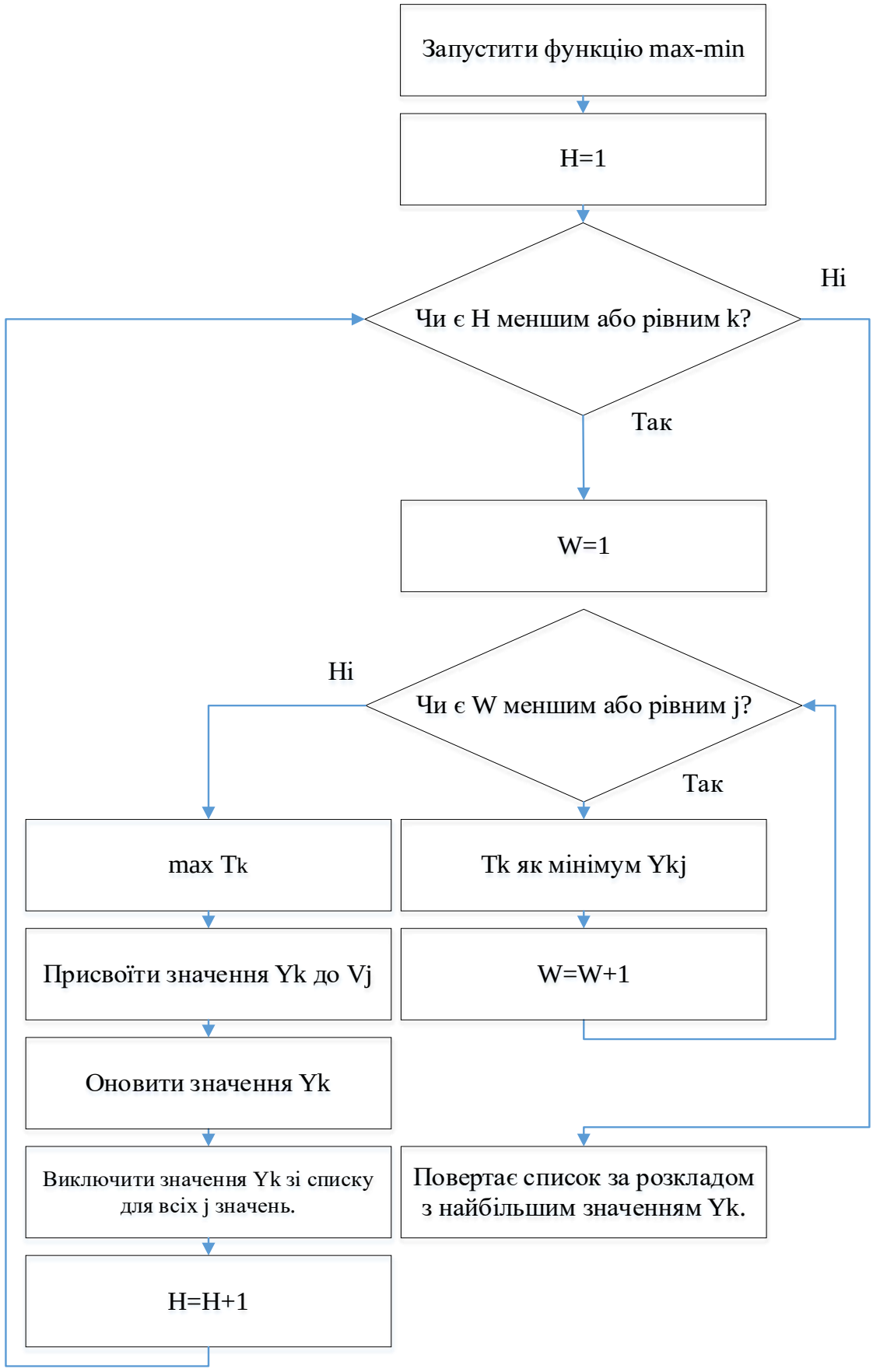


Рисунок 3.5. Порядок виконання кроків процедури мінімізації ресурсів для обслуговування всіх запитів за допомогою методу «max-min»

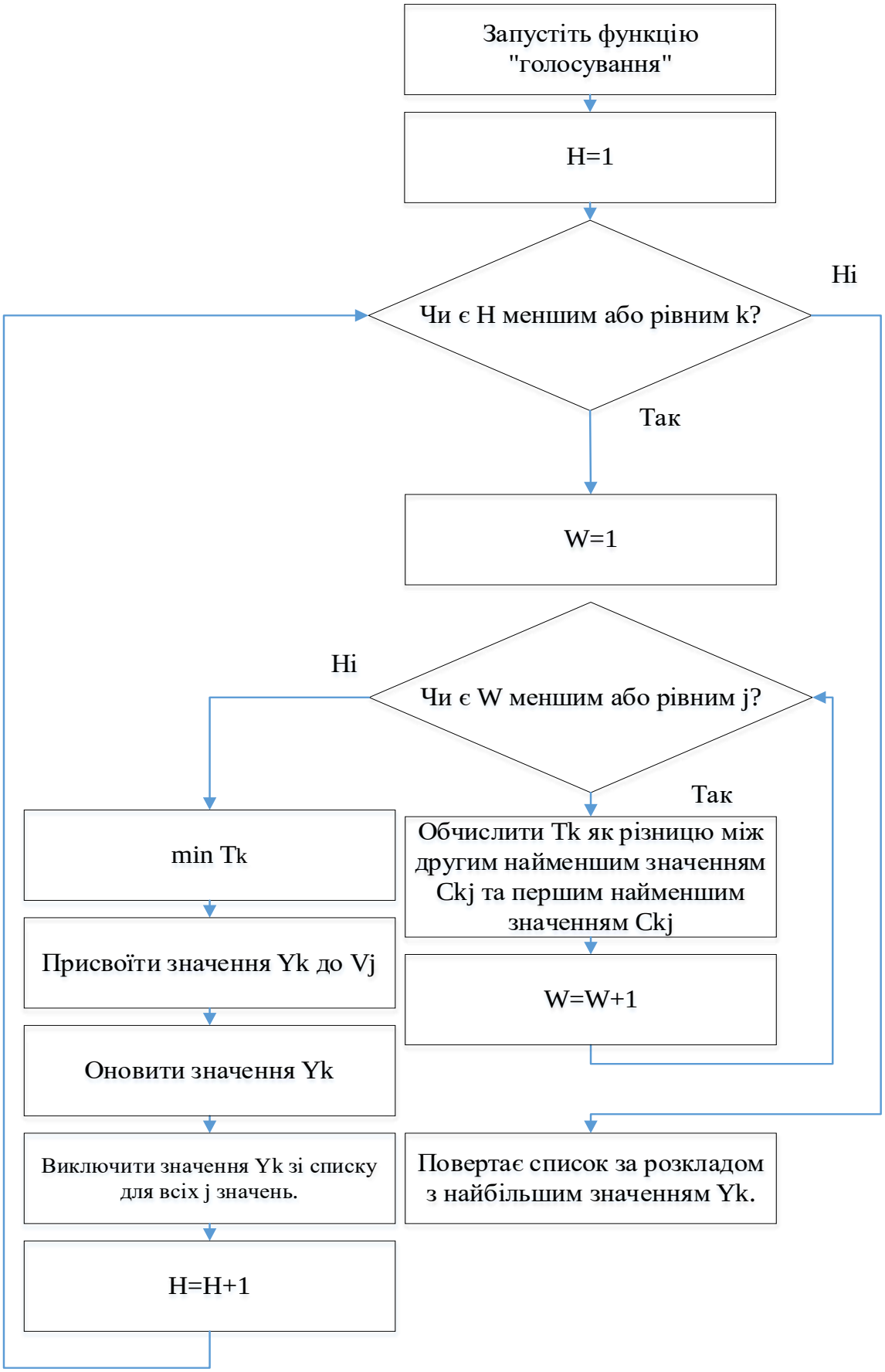


Рисунок 3.6 – Порядок виконання кроків процедури мінімізації ресурсів для обслуговування всіх запитів за допомогою генетичного методу

Метод багатокритеріального розподілу ґрунтується на побудові багатокритеріальної функції з мінімізацією кількості використаних фізичних машин $\min\left(\sum_{S} x_{i,j}\right)$ та мінімізацією загальної відповідності продуктивності ресурсів між віртуальними та фізичними машинами $\min\left(\sum_{S} WV_{i,j}\right)$, де $x_{i,j}$ позначає елемент відображення між віртуальною машиною v_i та фізичною машиною p_j [15, 16]. Якщо віртуальна машина v_i розміщена на фізичній машині p_j, то $x_{i,j}$ дорівнює 1. В іншому випадку $x_{i,j}$ дорівнює 0.

Отже, вираз $\sum_{S} x_{i,j}$ формалізує загальну кількість використаних фізичних машин у рамках рішення *S*.

У виразі для відповідності продуктивності ресурсів

$$WV_{i,j} = \sqrt{\sum_{k=1}^{4}\left(y \times v_{i,k} - y \times p_{j,k}\right)^2} \qquad (3.17)$$

використовується нормалізований показник продуктивності віртуальної машини v_i – $y \times v_{i,k}$, та відповідний показник нормалізації продуктивності фізичної машини p_j – $y \times p_{j,k}$ (y – коефіцієнт нормалізації).

Також у даному виразі k=1, 2, 3 формулює наявність ресурсів ЦП, пам'яті, диска, системи захисту (файерволу) відповідно.

Для усунення недоліків, пов'язаних з неефективним використанням фізичних ресурсів [10], пропонується вдосконалити метод розподілу ресурсів на основі передбачення запитів віртуальних машин. Якщо співвідношення різних типів ресурсів у запиті віртуальної машини ближче до доступних ресурсів фізичної машини, тобто, чим ближче співвідношення ресурсів v_{i1}:v_{i2}:v_{i3}:v_{i4} віртуальної машини v_i до p_{j1}:p_{j2}:p_{j3}:p_{j4} фізичної машини p_j, тим менше ймовірність втрат ресурсів для даної фізичної машини [11].

Тут v_{i1}, v_{i2} та v_{i3} представляють запитану кількість ядер ЦП, об'єм пам'яті та розмір диска віртуальної машини v_i відповідно; а p_{j1}, p_{j2}, p_{j3} та p_{j4} позначають доступну кількість ядер ЦП, об'єм пам'яті, розмір диска та ресурс файерволу фізичної машини p_j відповідно.

Отже, створюється модель відповідності пропорцій ресурсів:

$$PWV_{i,j} = \sqrt{\sum_{k=1}^{4}\left(\left(\frac{y \times p_{j,k} \times v_{i,1}}{p_{j,1}} - y \times v_{i,k}\right)R_k\right)^2}, \qquad (3.18)$$

де p_{jk} – доступна ємність ресурсу типу k для фізичної машини p_j;

v_{ik} запитана ємність ресурсу віртуальної машини v_i.

R_k – коефіцієнт балансування, який регулює значення комплексного параметру $\left(\frac{y \times p_{j,k} \times v_{i,1}}{p_{j,1}} - y \times v_{i,k}\right)$ для різних типів ресурсів.

Наприклад, якщо результат рішення виразу $\left(\frac{y \times p_{j,k} \times v_{i,1}}{p_{j,1}} - y \times v_{i,k}\right)$ для ресурсів пам'яті та файерволу становлять 1 та 100 відповідно, врахування ресурсів файерволу стає більш вагомим фактором.

Тому R_k для ресурсів файерволу повинен бути встановлений нижче, ніж для ресурсів пам'яті, наприклад, R_k = 1 для ресурсів пам'яті та R_k = 0.1 для ресурсів файерволу [12].

Таким чином, ми формулюємо багатоцільову задачу оптимізації ресурсів з урахування ризиків кібербезпеки наступним чином.

Для виділення на основі кількості використовуваних віртуальних машин $\sum_S x_{i,j}$, загальної відстані між ресурсами віртуальних та фізичних машин $\sum_S WV_{i,j}$ і загальної відстані між частками ресурсів $\sum_S PWV_{i,j}$ потрібно забезпечити [22]:

$$\min\left(\sum_{S} x_{i,j}\right); \tag{3.19}$$

$$\min\left(\sum_{S} WV_{i,j}\right), \tag{3.20}$$

$$\min\left(\sum_{S} PWV_{i,j}\right). \tag{3.21}$$

Первинна мета багатоцільової задачі оптимізації (3.19) для виділення ресурсів – мінімізувати загальну кількість використовуваних фізичних машин [23]. Ця мета залежить від значень окремих елементів відображення x_{ij} між віртуальною машиною v_i та фізичною машиною p_j в межах рішення S [14, 17].

Друга мета задачі (3.20) – мінімізувати загальну відстань між ресурсами віртуальних машин та фізичних машин в межах рішення S. Ця мета залежить від відстані між ресурсами $\sum_{S} WV_{i,j}$ між віртуальною машиною v_i та фізичною машиною p_j [19].

Третя мета задачі (3.21) – мінімізувати загальну відстань між частками ресурсів віртуальних машин та фізичних машин в межах рішення S. Ця мета базується на загальній відстані між частками ресурсів $PWV_{i,j}$ між віртуальною машиною v_i та фізичною машиною p_j [18].

Загальні ресурси процесорів, пам'яті, дискові ємності та ресурси файерволу, що запитані віртуальними машинами, розміщеними на фізичній машині p_j, менше, ніж доступні ресурси $p_{j\text{доп}}$. Тому обмеження оптимізаційної задачі можна сформулювати наступним чином [35]:

$$\sum_{S} v_{i,j} \times x_{i,j} \le p_{j,1}; \tag{3.22}$$

$$\sum_{S} v_{i,2} \times x_{i,j} \le p_{j,2}, \tag{3.23}$$

$$\sum_{S} v_{i,3} \times x_{i,j} \le p_{j,3}. \tag{3.24}$$

$$\sum_{S} v_{i,4} \times x_{i,j} \le p_{j,4}. \tag{3.25}$$

Наступним завданням оптимізації є покращення алгоритму розв'язання для прискорення швидкості розв'язання багатоцільової оптимізаційної функції. Для цього використаємо класичний алгоритм для розв'язання багатоцільової задачі оптимізації – NSGA-II [66].

NSGA-II (Nondominated Sorting Genetic Algorithm II) – це еволюційний алгоритм оптимізації, який використовується для вирішення багатокритеріальних задач. Цей алгоритм відноситься до сім'ї генетичних алгоритмів і призначений для вирішення задач, де існують кілька конфліктуючих цілей, які потрібно оптимізувати одночасно.

NSGA-II базується на ідеї ранжування недомінованих рішень (тобто рішень, які не можуть бути покращені в одній цілі без погіршення в інших цілях) і здійснює розподіл популяції на фронти Парето (множину недомінованих рішень).

Основна мета NSGA-II – знайти оптимальну апроксимацію фронту Парето, тобто набір рішень, які найкраще відображають різні трейд-офи між конфліктуючими цілями.

Основні кроки роботи NSGA-II включають генерацію початкової популяції, застосування операторів схрещування та мутації для створення нових особин, ранжування рішень за допомогою недомінованого сортування та ранжування критеріїв, вибірку недомінованих рішень для наступної популяції, а також застосування архіву для зберігання недомінованих рішень та їх альтернативних розподілів.

Як алгоритм багатоцільового сортування генетичного алгоритму, цей алгоритм широко використовується для розв'язання завдань багатоцільової оптимізації та демонструє добру ефективність.

Однак у алгоритму NSGA-II є недолік: час обчислення значень придатності (тобто цільових функцій) часто є великим, що може загрожувати своєчасності виділення ресурсів.

Крім того, необхідно обчислити значення придатності для великої кількості особин в еволюції популяції.

Таким чином, ми пропонуємо покращити алгоритм NSGA-II для прискорення процесу розв'язання шляхом паралельного обчислення функції придатності.

Були використані багатоядерні процесори для паралельного обчислення значень придатності для особин, що прискорює збіжність запропонованого алгоритму [82, 83, 85].

Значення придатності для кожної розглядаємої особини обчислюються наступним чином:

$$f_1(E_k) = \sum_S x_{i,j}; \tag{3.26}$$

$$f_2(E_k) = \sum_S WV_{i,j}, \tag{3.27}$$

$$f_3(E_k) = \sum_S PWV_{i,j}. \tag{3.28}$$

Таким чином, третя частина моделювання присвячена формалізації та кодуванню даних в рамках генетичного алгоритму для балансування навантаження ресурсів.

Блок схема алгоритму представлена на рис. 3.8.

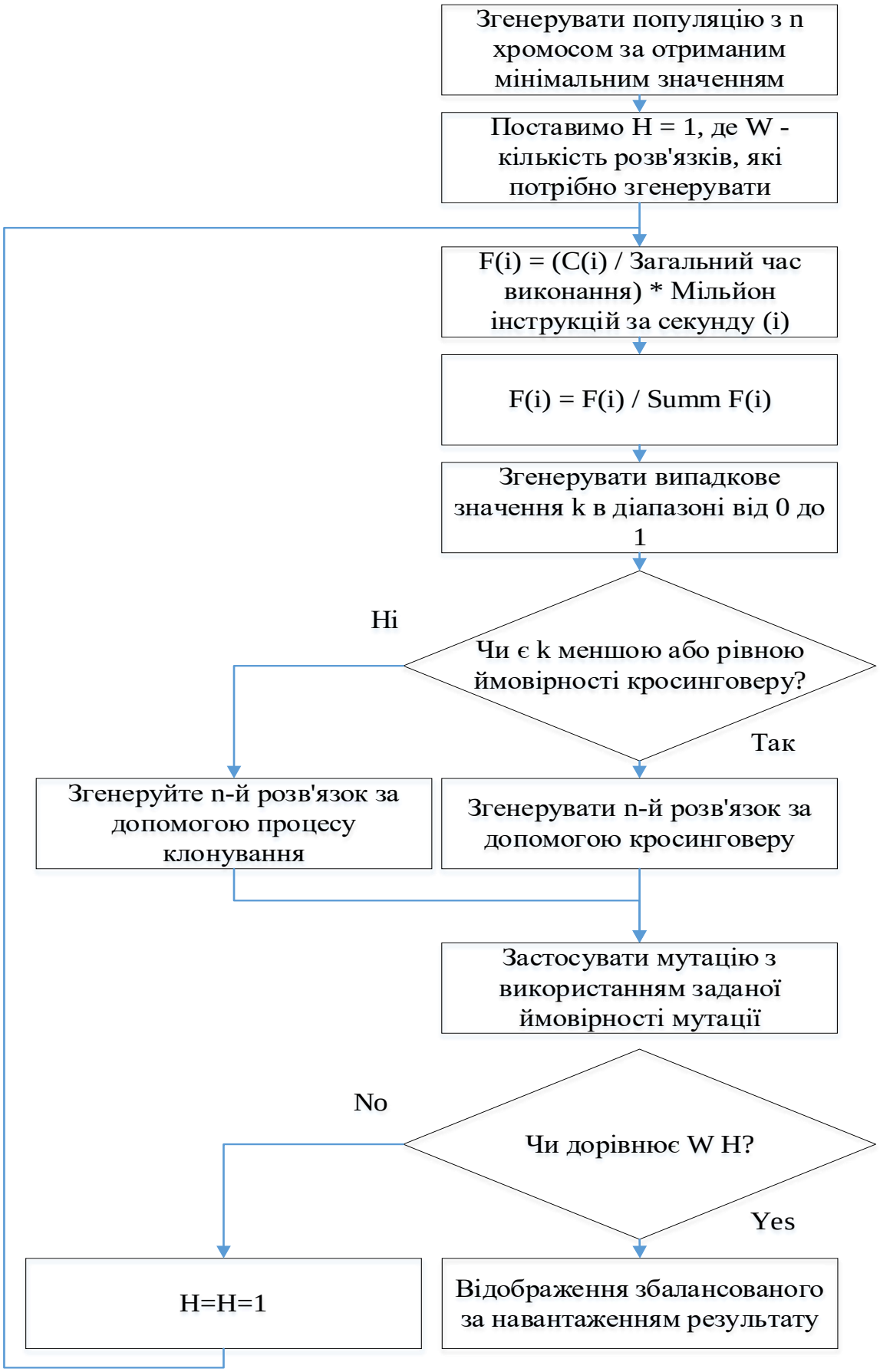


Рисунок 3.8 – Порядок виконання кроків генетичного алгоритму балансування навантаження ресурсів

Проведені порівняльні дослідження запропонованого методу з методами, що базуються на алгоритмах SPEA2 та NSGA-II показали, що з чотирма потоками метод SPEA2 досягає використання ЦП на рівні 59% та пам'яті на рівні 61%, NSGA-II досягає використання ЦП на рівні 64% та пам'яті на рівні 66%, тоді як запропонований метод досягає використання ЦП на рівні 63% та пам'яті на рівні 65%.

Отже, запропонований метод превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», який враховує результати аналізу попередніх даних та базується на моделі багатоцільового розподілу хмарних ресурсів, дозволяє завчасно провести прогнозування завантаженості фізичних пристроїв хмарного середовища та запобігти втратам обчислювального ресурсу.

Сформовані можливі черги запитів на найбільш витратний за часом формування хмарний ресурс – віртуальні машини, мають відхилення від реальних запитів не більше 15%.

Комплексне використання запропонованих методів дозволило зменшити затримку в обслуговуванні обчислювальних ресурсів до 5% за рахунок підвищення рівень балансування завантаження обчислювальних ресурсів хмарного середовища.

3.6. Висновки за розділом 3

Узагальнені вимоги до методу розподілу хмарних ресурсів при використанні моделі обслуговування «Інфраструктура у якості сервісу». При цьому особливо відмічені такі вимоги: швидка обробка багатьох динамічних запитів до ресурсів; надання оптимальних ресурсів для задоволення запитів; мінімізація використання фізичних серверів та рівномірний розподіл різних типів ресурсів для зниження втрат ресурсів.

Запропонована узагальнена структура методу адаптивного розподілу хмарних ресурсів. Розглянуті основні складові та етапи реалізації даного методу, визначені особливості кожного з етапів та механізми взаємодії між ними.

Розглянутий підхід до адаптивного прогнозування завантаженості хмарних ресурсів на основі тесту на послідовність серій. Запропонований підхід дозволяє адаптивно вибирати найбільш відповідний метод прогнозування залежно від характеру даних та їхньої змінюваності, з урахуванням особливостей запитів до ресурсів. За рахунок такої додаткової процедури зменшується час прогнозування шляхом видобутку основних компонентних послідовностей до 20%.

Запропонований метод превентивного формування черг запитів на віртуальні машини хмарного середовища, який декомпозований на такі послідовні етапи: прогнозування майбутніх запитів на віртуальні машини; визначення необхідності заздалегідь виділяти ресурси; встановлення послідовності запитів на віртуальні машини. Даний метод враховує результати аналізу попередніх даних та базується на моделі багатоцільового розподілу хмарних ресурсів, що дозволяє завчасно провести прогнозування завантаженості фізичних пристроїв хмарного середовища та запобігти втратам обчислювального ресурсу. Його використання дозволило сформувати можливі черги запитів на найбільш витратний за часом формування хмарний ресурс – віртуальні машини, з відхиленням від реальних запитів не більше 15%.

Розглянута модель багатоцільового розподілу ресурсів, котра використовується як при превентивному формуванні черг запитів на віртуальні машини хмарного середовища, так і при адаптивному розподілі хмарних ресурсів. Послідовно при моделювання розглянуті такі задачі: мінімізація загальної кількості використовуваних фізичних машин; мінімізація загальної відстані між ресурсами віртуальних машин та фізичних машин за критерієм загальної відповідності продуктивності ресурсів між віртуальними та фізичними машинами; формалізація та кодування даних в рамках генетичного

алгоритму для балансування навантаження ресурсів. В результаті удосконалено метод адаптивного розподілу ресурсів хмарного середовища, який відрізняється від відомих використанням тестування на послідовність серій, математичного апарату удосконаленого генетичного алгоритму NSGA-II та результатами прогнозу запитів на віртуальні машини, що дозволило підвищити ефективність використання хмарних обчислювальних ресурсів за рахунок реалізації балансу між ресурсами центрального процесора та оперативної пам'яті та зменшення затримки в обслуговуванні хмарних ресурсів. Комплексне використання запропонованих методів дозволило зменшити затримку в обслуговуванні обчислювальних ресурсів в процесі функціонування наданої замовнику віртуальної інфраструктури до 5% за рахунок підвищення рівня балансування завантаження обчислювальних ресурсів хмарного середовища.

Основні результати розділу надруковані в наукових працях [3, 5, 6, 11, 12, 13] (додаток А).

РОЗДІЛ 4
ДОСЛІДЖЕННЯ ЗАПРОПОНОВАНИХ МЕТОДІВ РОЗПОДІЛУ РЕСУРСІВ В КОМП'ЮТЕРНИХ СИСТЕМАХ ПРИ НАДАННІ ХМАРНИХ ІНФРАСТРУКТУРНИХ ПОСЛУГ

4.1. Дослідження методу адаптивного прогнозування запитів на ресурси на основі тесту на послідовність серій

Виберемо дві часові послідовності та назвемо їх умовно P1 та O1. Ці послідовності містять данні про запити на віртуальні машини у кластері Alibaba [45, 46] для експериментального набору даних.

Alibaba Cluster Trace фіксує докладну статистику для спільних робочих завдань тривалого виконання і пакетних завдань протягом 24 годин. Трасу складається з трьох частин:

– статистика вивченого однорідного кластера з 1 313 машин, включаючи конфігурацію обладнання кожної машини та використання ресурсів (CPU, Memory, Disk) протягом 12 годин (друга половина 24-годинного періоду);

– навантаження завдань тривалого виконання, включаючи трасу всіх запитів та дій з розгортання контейнерів, а також трасу використання ресурсів протягом 12 годин;

– навантаження пакетних завдань, включаючи трасу всіх запитів і дій пакетних завдань, а також трасу використання ресурсів для кожного екземпляра протягом 24 годин [46].

Також існує друга версія траси cluster-trace-v2018, яка включає приблизно 4 000 машин протягом 8 днів. Окрім більшого масштабу в порівнянні з trace-v2017, ця частина траси також містить інформацію про DAG (Directed Acyclic Graph – ациклічний направлений граф) для виробничих пакетних завдань [46].

Послідовності P1 та O1 включають лише дані про використання процесора і пам'яті. Використовуватимемо послідовність P1 для ілюстрації

процесу адаптивного прогнозування. Послідовність P1 складається з 74 точок вимірювання (приблизно 450 хвилин) і містить 15 типів ВМ. Кожна точка вимірювання відображає загальну кількість віртуальних машин протягом 5-хвилинного інтервалу. Застосовуємо метод аналізу основних компонент для виділення компонентних послідовностей P2 і P3 та розраховуємо поріг T_{td}=85% відповідно до заданих порогів ε_h=5% і ε_t=20%.

Графіки цих послідовностей подані на рис. 1.

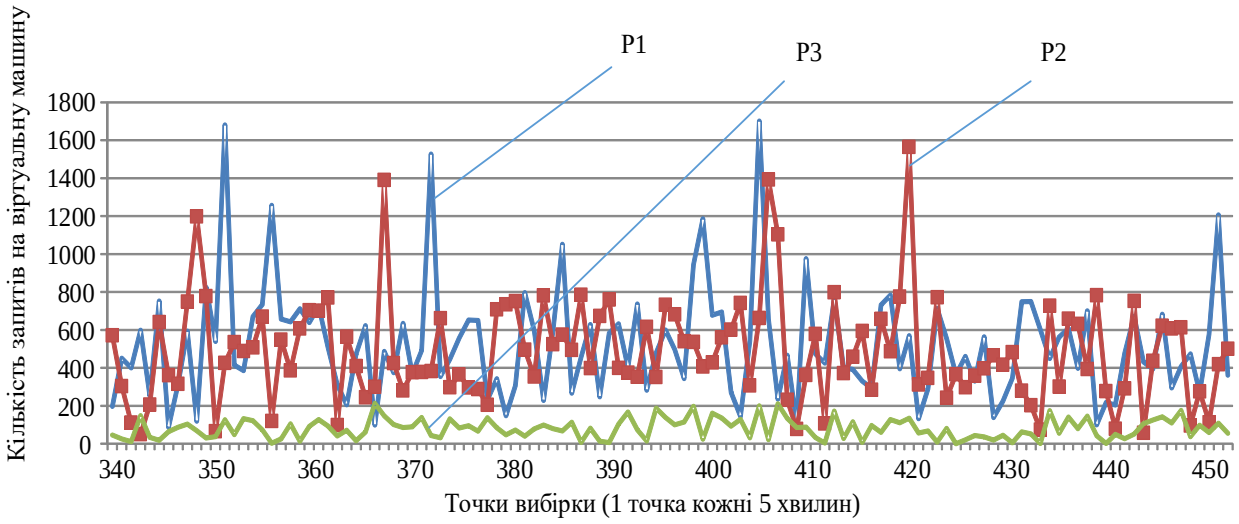


Рисунок 4.1 – Кількість нових віртуальних машин,
створених для онлайн-сервісів

На рис. 4.1 графік P1 відображає загальну кількість 28 різних типів нових віртуальних машин.

Криві P2 і P3 показують кількість нових віртуальних машин для типів з процесором CPU = 400 і об'ємом пам'яті RAM = 1.56, а також для типів з процесором CPU = 800 і об'ємом пам'яті RAM = 3.13, відповідно.

Послідовності P2 та P3 представляють кількість віртуальних машин для типів із 4-ядерним CPU і 1,56 пам'яті (CPU = 400, mem = 1,56) та 8-ядерним CPU і 3,13 пам'яті (CPU = 800, mem = 3,13) відповідно.

Потрібно відмітити, що кількість ядер процесора і обсяг пам'яті є нормалізованими. Можливо побачити динамічні зміни кількості запитів на

віртуальних машин, які проявляють характер раптових змін, що ускладнює передбачення майбутніх ресурсних запитів.

Також бачимо, що послідовність P2 з 4-ядерним CPU і 1,56 пам'яті відповідає тренду послідовності P1. Послідовність P3 з 8-ядерним CPU і 3,13 пам'яті в цілому співпадає з трендом послідовності P1, але має деякі різниці в деталях коливань.

Використаємо метод міжквартильного розмаху (IQR) [60] для виявлення викидів, які позначені знаками " + " на рис. 4.2.

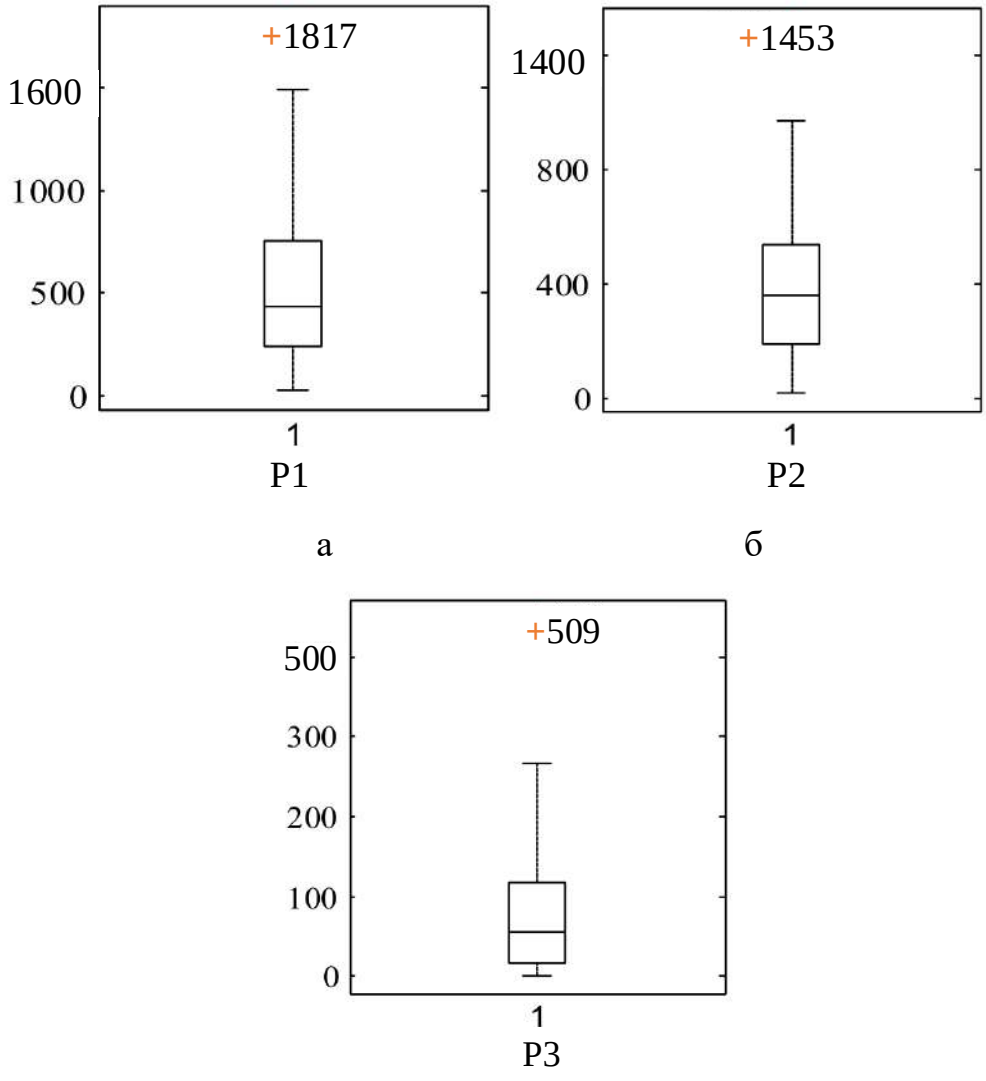


Рисунок 4.2 – Діаграми з віконними послідовностями запитів до віртуальних машин. Перший (а), другий (б) і третій (в) графіки показують викиди зі знаками "+" для послідовностей P1, P2 і P3 відповідно

Міжквартильний розмах (IQR) – це статистична міра розподілу даних, яка використовується для визначення варіації або розкиду в наборі даних. Міжквартильний розмах IQR обчислюється за допомогою квартилів, а саме третього (Q3) і першого (Q1) квартилів. Він вказує на ширину центральної "половини" даних, тобто інтерквартильний діапазон між 25% (Q1) та 75% (Q3) квартилями набору даних.

Основні кроки для обчислення IQR такі [117]:

– Сортування даних: Дані сортуються у порядку зростання або спадання.

– Знаходження Q1 і Q3:

– Перший квартиль (Q1) - це значення, яке розділяє найнижчі 25% даних від інших 75%.

– Третій квартиль (Q3) - це значення, яке розділяє найнижчі 75% даних від інших 25%.

– Обчислення IQR:

Міжквартильний розмах IQR розраховується як різниця між Q3 і Q1:

$$IQR = Q3 – Q1. \quad (4.1)$$

Зазвичай вважається, що дані, які знаходяться за межами діапазону Q1 - 1.5 * IQR до Q3 + 1.5 * IQR, є "нормальними" або не викидами [2, 6]. Всі дані, що виходять за ці межі, можуть бути викидами або аномаліями.

Міжквартильний розмах IQR є важливою мірою в аналізі даних та виявленні аномалій, оскільки він дозволяє виділяти центральну частину набору даних, викидаючи значення, які можуть бути спричинені випадковими або нетиповими подіями [93, 105]. Це допомагає покращити розуміння розподілу даних і забезпечити більш точний аналіз.

Таким чином, використовуючи метод міжквартильного розмаху отримуємо передбачені послідовності, які подані на рис. 4.3.

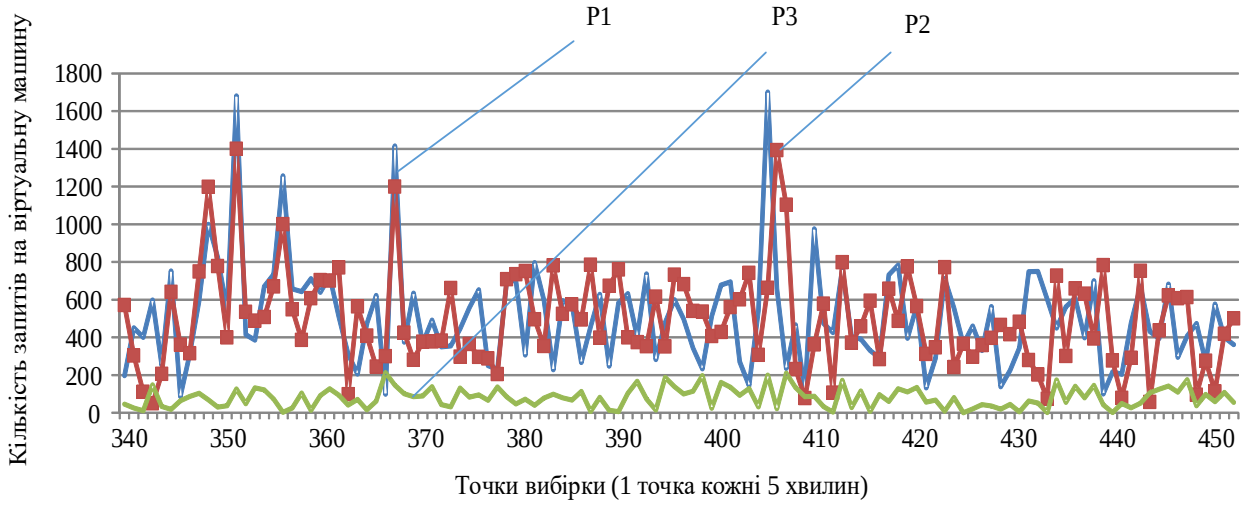


Рисунок 4.3 – Попередньо оброблені послідовності
запитів віртуальної машини

На графіках рис. 4.3 криві P1, P2 і P3 представляють попередньо оброблені послідовності після використання методу міжквартильного розмаху для заміни динамічних виплесків вихідних послідовностей P1, P2 і P3 відповідно.

Наступним кроком у дослідженні є застосування адаптивного методу прогнозування на основі тесту на послідовність серій для прогнозування цих передпроцесних послідовностей, що наведені на рис. 4.3.

Спочатку ми обчислюємо значення остатніх тенденцій (RT) для цих послідовностей. Порогове значення, R_{td}, встановлено на рівні 25, що було визначено в ході експериментального тестування на даних кластера Alibaba [46]. Важливо зауважити, що значення R_{td} може відрізнятися для різних наборів даних або сценаріїв. Вибір значення R_{td} залежить від числа факторів, таких як важливість вхідних вимог до безпеки, надійності та достовірності. У роботі значення R_{td} було встановлено на основі експериментальних досліджень з використанням досвіду експертів з компанії LineUp.

У дослідженні ми використовуємо перші 75 точок даних як навчальний набір, а потім по 7, 9 і 14 точок даних відповідно як тестовий набір. Якщо

значення RT послідовності опускається нижче передвстановленого порогу Rtd, ми використовуємо метод EEMD-RT-ARIMA [57] для прогнозування. В іншому випадку, якщо значення RT перевищує поріг, ми вибираємо метод EEMD-ARIMA[58].

На рис. 4.4 показано середню абсолютну відсоткову похибку результатів прогнозування.

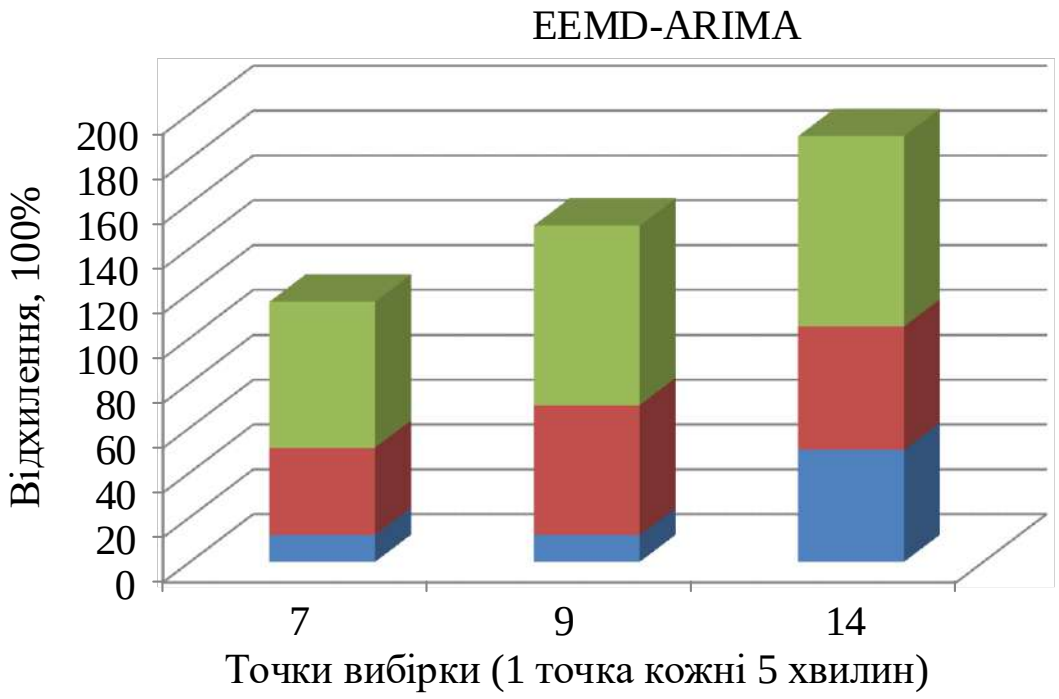
Очевидно, що значення абсолютної відсоткової похибки значно зростають при прогнозуванні на 9 і 14 точок порівняно з прогнозуванням на 7 точок. Наприклад, метод EEMD-RT-ARIMA досягає відхилення 10% при прогнозуванні на 7 точок для послідовності P1, тоді як для прогнозування на 9 і 14 точок відхилення становить відповідно 30% і 55%.

Так само метод EEMD-ARIMA досягає відхилення 12% при прогнозуванні на 7 точок для послідовності P2, але реєструє відхилення 39% і 65% при прогнозуванні на 9 і 14 точок відповідно. Це свідчить про те, що обидва методи більш підходять для короткострокового прогнозування через стрімку зміну даних на короткий період часу.

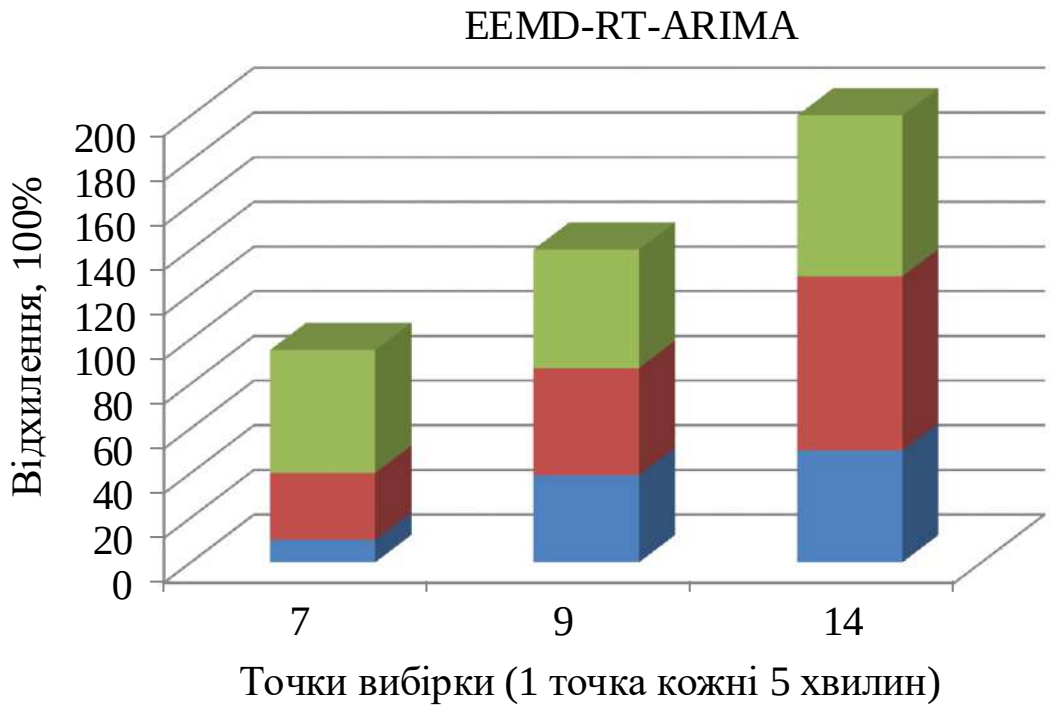
Вибір між двома методами залежить від характеристик даних. Загалом метод EEMD-RT-ARIMA переважає метод EEMD-ARIMA для послідовностей P1 і P3, коли їхні значення відхилень становлять відповідно 20 і 23. З іншого боку, для послідовності P2 з відхиленнями в 20, метод EEMD-ARIMA забезпечує вищу точність результатів. Це підкреслює ефективність запропонованого методу прогнозування.

На практиці, у випадках, коли послідовність даних має динамічні зміни, кумулятивна похибка прогнозу компонентних послідовностей, отриманих з використанням розкладання EEMD, може бути меншою, ніж похибка, викликана нестаціонарною послідовністю.

У результаті метод EEMD-ARIMA забезпечує вищу точність прогнозу в таких випадках. Навпаки, метод EEMD-RT-ARIMA більш ефективно зменшує накопичення похибок прогнозу для послідовностей з меншою флуктуацією, що призводить до вищої точності прогнозу.



а



б

Рисунок 4.4 – Діаграми відхилення (%) для вхідних даних кластера Alibaba у випадку адаптивного прогнозування для різних типів послідовностей запитів на віртуальній машині

На рис. 4.5-4.7 показані майбутні значення на 5 точок, прогнозовані за допомогою методу прогнозування запитів на ресурси на основі тесту на послідовність серій.

На графіках рис. 4.5-4.7 зображені результати порівняння фактичних значень з прогнозованими 5-бальними значеннями, отриманими за допомогою запропонованого методу прогнозування запитів на ресурси на основі тесту на послідовність серій для попередньо оброблених послідовностей P1, P2 та P3 відповідно.

Як видно з цих графіків використання запропонованого методу прогнозування запитів на ресурси на основі тесту на послідовність серій знижує загальну кількість помилок прогнозу у більшості практичних випадків.

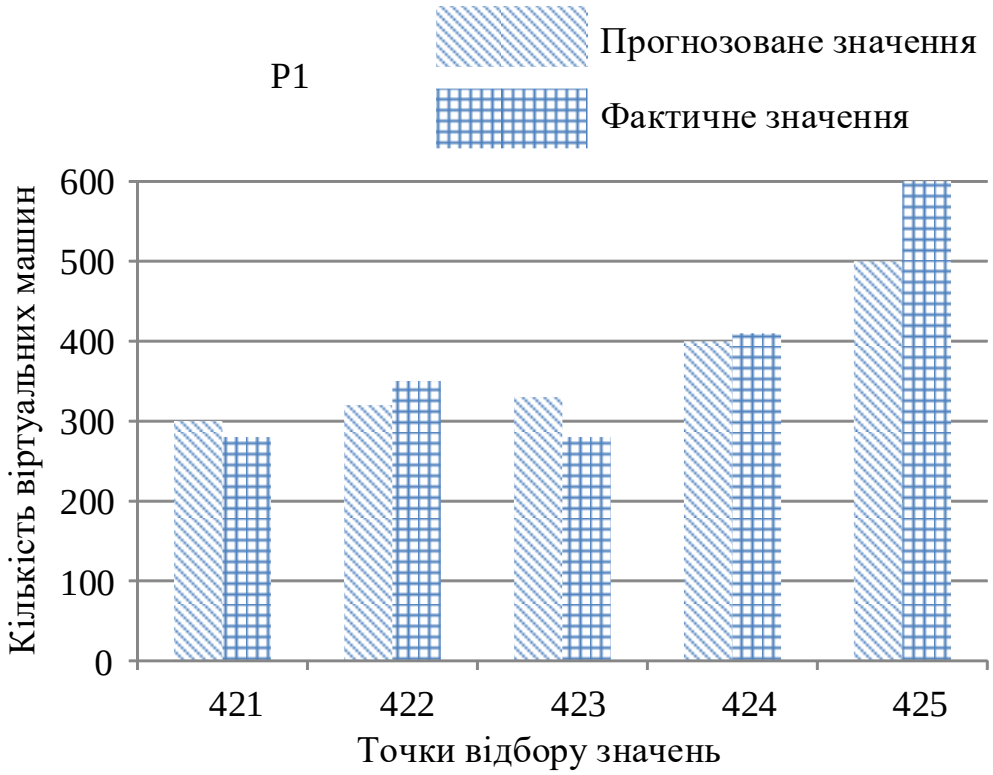


Рисунок 4.5 – Кількість передбачених віртуальних машин для послідовності P1

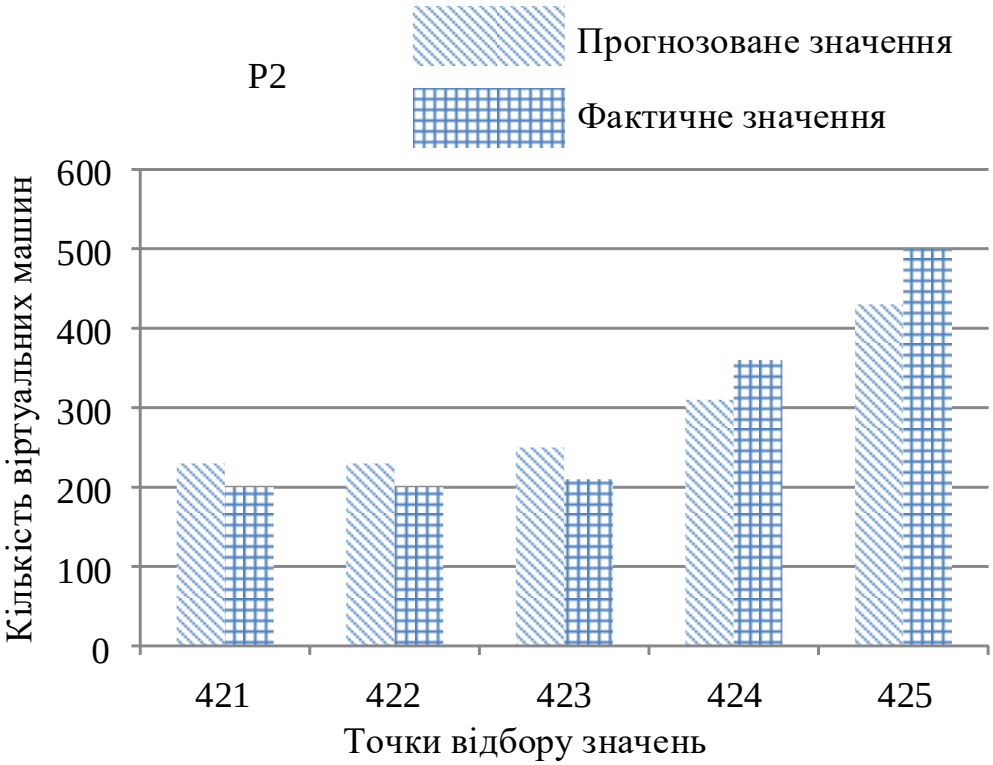


Рисунок 4.6 – Кількість передбачених віртуальних машин для послідовності P2

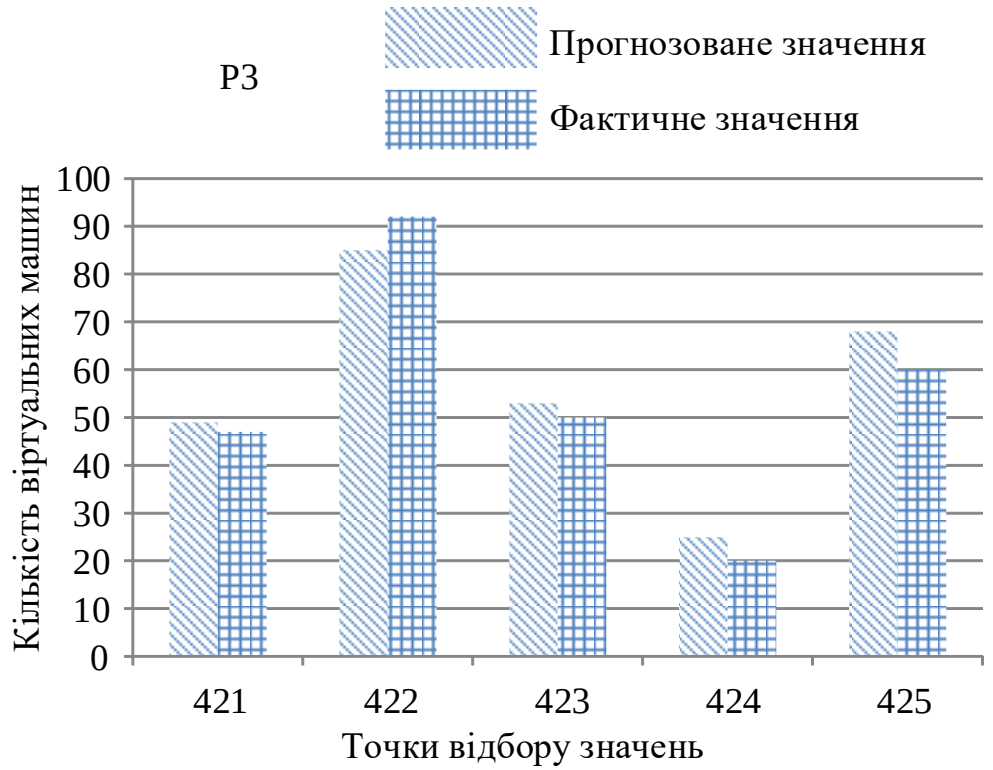


Рисунок 4.7 – Кількість передбачених віртуальних машин для послідовності P3

Аналогічно, зроблено прогноз значень на 5 точок для послідовності О1, як показано на рис. 4.8-4.10.

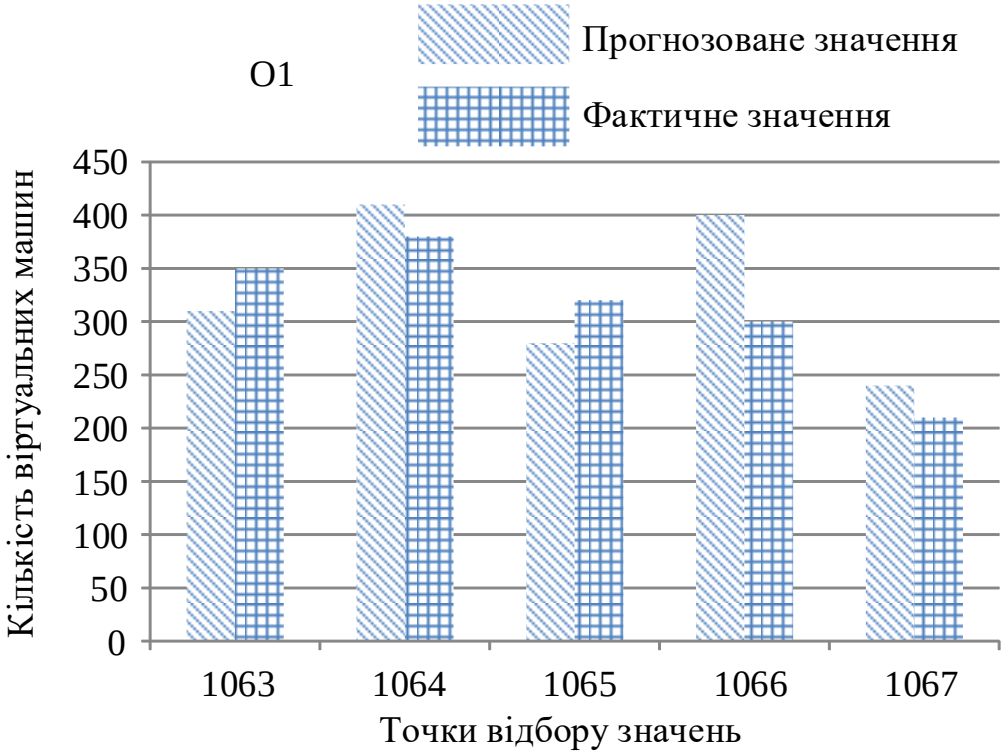


Рисунок 4.8 – Кількість передбачених віртуальних машин для послідовності О1

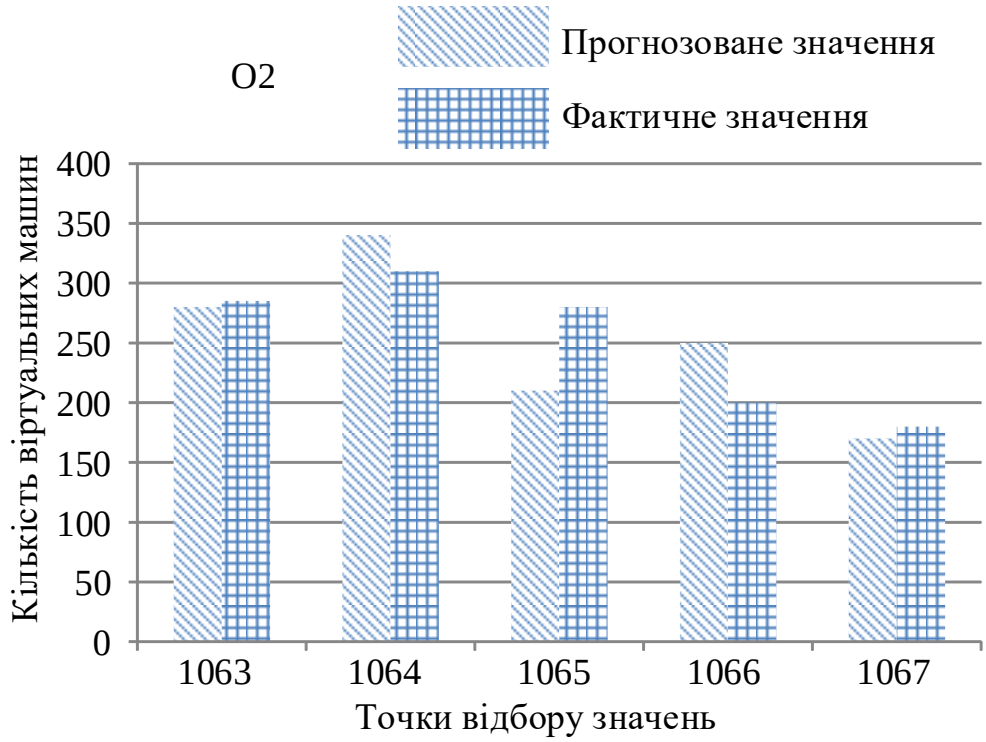


Рисунок 4.9 – Кількість передбачених віртуальних машин для послідовності О2

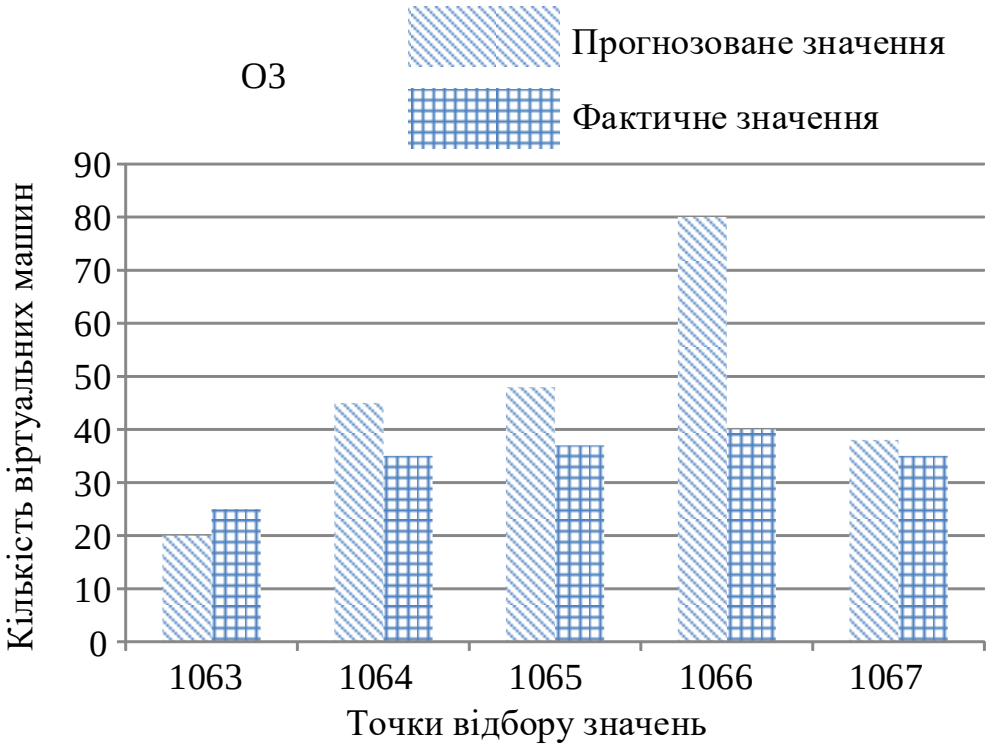


Рисунок 4.10 – Кількість передбачених віртуальних машин для послідовності O3

На графіках рис. 4.8-4.10 зображені результати порівняння фактичних значень і прогнозованих 5-точкових значень, отриманих за допомогою запропонованого методу прогнозування запитів на ресурси на основі тесту на послідовність серій для попередньо оброблених послідовностей O1, O2 і O3, відповідно. Так як і в попередньому випадку можемо спостерігати незначні помилки прогнозу у випадку використання методу прогнозування запитів на ресурси на основі тесту на послідовність серій.

4.2. Дослідження методу адаптивного розподілу хмарних ресурсів

Як показано на рисунках 4.5-4.7, для 425 точок вибірки передбачається 519 віртуальних машин (4-ядерних процесорів та 1,56 пам'яті) та 62 віртуальні машини (8-ядерних процесорів та 3,13 пам'яті). Варто відзначити, що спостерігається раптовий ріст віртуальних машин після 300 точок. Тому

раціонально виділяти ресурси для деяких віртуальних машин заздалегідь, щоб зменшити час розподілу ресурсів на 424 точках вибірки.

Проведені дослідження показали, що при встановленні співвідношення проактивного виділення ресурсів $P_i = 0,3$ можна досягти позитивного ефекту.

Кількість віртуальних машин, які потрібно створити на 424 точках вибірки, можна розрахувати за формулою $408 + (457 + 62)* P_i = 564$.

Кількість реальних доступних обчислювальних засобів становить 2972. Проблема розподілу ресурсів полягає в створенні 564 віртуальних машин на 2972 доступних обчислювальних засобах.

Також на рис. 4.8-4.10 можна побачити, що за допомогою запропонованого методу можна передбачити 281 віртуальну машину (4-ядерних процесорів та 1,56 пам'яті) та 31 віртуальну машину (8-ядерних процесорів та 3,13 пам'яті) для 1065 точок вибірки для послідовності O1.

Якщо встановити співвідношення проактивного виділення ресурсів $P_i = 0,3$, то необхідно передбачити $423 + (281 + 31) * P_i = 527$ віртуальних машин на 2972 доступних обчислювальних засобах на 1064 точках вибірки.

Навіть якщо передбачення може бути неточним, проактивне виділення ресурсів в незначній мірі перевищить необхідний поріг.

Наприклад, якщо передбачається 893 або 297 запитів на віртуальну машину для 425 точок вибірки, середньо абсолютна похибка відсотка перевищить 50%. Це означає, що передбачення є неточним. Нам потрібно створити на 268 або 89 віртуальних машин більше, ніж вихідна кількість 408 згідно зі стратегією проактивного виділення ресурсів для 424 точок вибірки.

У вказаному випадку загальний обсяг використання ресурсів все одно буде меншим, ніж фактична кількість запитів на віртуальні машини для 425 точок вибірки.

Проте, чим більше проактивне число запитів на віртуальні машини, тим довше час розподілу ресурсів для 424 точок вибірки. Тому помилку передбачення слід обмежити в певних межах.

Для оцінки ефективності розробленого в дисертації методу використовуємо наступні метрики для порівняння нашого методу з іншими [53, 54].

1. Кількість використовуваних обчислювальних ресурсів N_{res}: Якщо кількість використовуваних обчислювальних ресурсів менша, то можна вимкнути деякі простоюючі обчислювальні ресурси для зменшення споживання енергії та витрат.

2. Відстань відповідності продуктивності ресурсів S_{prod}: Чим менше відстань, визначена за опомогою відповідності продуктивності ресурсів, тим краще віртуальні машини відповідають доступним обчислювальним ресурсам за продуктивністю.

3. Відстань відповідності частки ресурсів S_{sub}: Чим менше відстань відповідності частки ресурсів, тим менше ресурсів втрачається.

4. Використання ресурсів L_{res}: Ефективний метод виділення ресурсів повинен максимізувати та узгоджувати використання кожного типу ресурсів.

5. Динаміка ресурсів T_{res}: Запропонований метод виділення ресурсів на основі прогнозу скорочує час створення віртуальних машин за рахунок виділення ресурсів для майбутніх запитів на віртуальні машини наперед. Чим менше часу вирішення, тим більше можна зекономити часу на виділення ресурсів.

6. Відмовостійкість P_{rej}: Вимірює, наскільки метод відмічається стійкістю до відмов обчислювальних вузлів або інших помилок в системі.

7. Забезпечення безпеки P_{sec}: Як вдало метод забезпечує захист даних і ресурсів від несанкціонованого доступу або атак.

Крім того, в симуляції використовувалася ймовірність змін, обернена до кількості змінних. Максимальна кількість оцінок значень пристосованості та максимальна кількість ітерацій для змін були встановлені на рівні 20000 і 100 відповідно.

Для налаштування імітаційної моделі було встановлено такі параметри:

розмір зміни – 200,

ймовірність комбінацій – 0,85,

індекс розподілу комбінацій – 20,

індекс розподілу змін – 20.

Крім того, в симуляції використовувалася ймовірність зміни, обернена до кількості змінних. Максимальна кількість оцінок значення адаптивності та максимальна кількість ітерацій для змін була встановлена на рівні 20000 та 100.

Для оцінки ефективності запропонованого методу було проведено порівняння з методом кругового перебору (RR), SPEA2 та NSGA-II за такими параметрами:

кількість використаних RR;

відстань узгодження продуктивності ресурсу;

відстань узгодження частки ресурсу;

використання ресурсу;

час розв'язання.

SPEA2 – ще один репрезентативний метод еволюційного багатокритеріального пошуку [90], який може отримати кілька оптимальних за Парето рішень за один прогін і широко застосовується в різних галузях, ставши стандартом для порівняння ефективності багатокритеріальних методів еволюційного пошуку.

Кожен метод було виконано 10 разів і розраховано середні результати.

Результати експерименту представлені в табл. 4.1 і 4.2.

З таблиць 4.1 і 4.2 видно, що використання SPEA2, NSGA-II і запропонованого адаптивного методу розподілу ресурсів дозволяє використовувати різну кількість обчислювальних засобів.

Таблиця 4.1 – Експериментальні результати розподілу ресурсів для послідовності P1

Метрики	RR	SPEA2	NSGA-II	Адаптивний метод	
				8 потоків	10 потоків
N_{res}	564	477	473	460	462

S_{prod}	352	436	403	408	395
S_{sub}	19	11	12	12	13
L_{res}(CPU) %	75	59	64	63	65
L_{res}(memory) %	29	60	65	64	66
T_{res} с.	0,3	1593	1551	977	886
P_{rej}	0,7	0,8	0,9	0,9	0,9
P_{sec}	0,75	0,8	0,9	0,9	0,9

Таблиця 4.2 – Експериментальні результати розподілу ресурсів для послідовності O1

Метрики	RR	SPEA2	NSGA-II	Адаптивний метод	
				8 потоків	10 потоків
N_{res}	527	423	435	430	426
S_{prod}	339	291	295	292	304
S_{sub}	18	11	12	12	11
L_{res}(CPU) %	73	64	63	64	62
L_{res}(memory) %	28	65	65	66	63
T_{res} с.	0,3	1592	1375	821	777
P_{rej}	0,75	0,85	0,85	0,9	0,9
P_{sec}	0,8	0,85	0,85	0,9	0,9

Слід також зазначити, що запропонований метод дозволяє адаптувати кількість використовуваних обчислювачів до умов експерименту, наприклад, 460 та 462 обчислювачів.

Чим менша кількість використовуваних обчислювальних засобів, тим більша економія ресурсів.

Балансування використання процесора та пам'яті ПМ досягається шляхом узгодження розподілу ресурсів, що зменшує нераціональне використання ресурсів.

З табл. 4.1 і 4.2 також видно, що для паралельних обчислень з 8 потоками метод SPEA2 досягає 59% використання процесора і 60% використання пам'яті, метод NSGA-II – 64% і 65%, а метод адаптивного розподілу ресурсів – 63% і 64%.

Проведене дослідження показало значні переваги використання методу адаптивного розподілу ресурсів над методом RR. Цей метод використовує найбільшу кількість інструментів перерахунку завдяки механізму опитування. Також використання цього методу погіршує балансування використання ресурсів процесора на 75% та пам'яті на 29% через механізм опитування. Це призводить до великих втрат ресурсів.

Однак динамічність ресурсів також дозволяє використовувати цей метод для аналізу продуктивності, оскільки він використовує простий евристичний алгоритм для вирішення проблеми. Порівняно з методами SPEA2 та NSGA-II, запропонований метод адаптивного розподілу ресурсів потребує менше часу для вирішення багатокритеріальних задач.

Наприклад, методи SPEA2 та NSGA-II потребують 1593 та 1551 секунди для розв'язання багатокритеріальної задачі відповідно, в той час як розроблений метод потребує лише 886 секунд для її розв'язання в паралельних обчисленнях з 10 потоками. Ефективне передбачення значно скорочує час створення віртуальної машини. Це забезпечує швидкий та відмовостійкий розподіл ресурсів.

У свою чергу, вищезгадані переваги в оцінках також дозволили нам стверджувати про підвищення таких характеристик, як відмовостійкість та безпека [109, 110].

Дані, наведені в табл. 4.1 і 4.2, свідчать про збільшення відмовостійкості та безпеки до 1,2 разів.

4.3. Практичні рекомендації по використанню методу адаптивного розподілу ресурсів

Розроблений адаптивний метод розподілу ресурсів є потужним інструментом для оптимізації використання обчислювальних ресурсів у віртуалізованих середовищах. Щоб ефективно використовувати цей метод, ось деякі практичні рекомендації:

У результаті досліджень сформульовані певні рекомендації: Перед використанням методу розподілу ресурсів рекомендується аналізувати історичні дані про використання ресурсів і запити на віртуальні машини. Це допоможе зрозуміти зміни в попиті та прогнозувати їх.

Налаштування параметрів: Для оптимальних результатів налаштуйте параметри адаптивного методу відповідно до потреб вашої організації. Це може включати встановлення розміру змін, ймовірності комбінацій та інших параметрів відповідно до вашого досвіду та потреб.

Моніторинг ресурсів: Постійно відстежуйте використання ресурсів у реальному часі. Якщо ви помічаєте зміни в попиті, адаптуйте розподіл ресурсів відповідно.

Резервні ресурси: Рекомендується мати деяку кількість резервних ресурсів, які можна виділити при несподіваному зростанні попиту. Адаптивний метод допоможе ефективно розподілити ці резерви.

Навчання моделей: Можливо, знадобиться тренувати моделі машинного навчання на історичних даних для більш точних прогнозів попиту на ресурси.

Захист від помилок прогнозування: Помітне зростання попиту може призвести до неточних прогнозів. Тому важливо мати стратегії резервування ресурсів для уникнення серйозних збоїв.

Постійне оновлення: Спостереження за тенденціями використання ресурсів і постійне оновлення параметрів методу дозволить вам завжди використовувати ресурси максимально ефективно.

Тестування та валідація: Перед впровадженням виробництва рекомендується провести тестування та валідацію розробленого методу на тестових даних, щоб переконатися в його ефективності.

Документація та навчання персоналу: Забезпечте належну документацію щодо використання методу і навчіть вашу команду, як ним користуватися.

Постійне вдосконалення: Технологічний ландшафт постійно змінюється, тому методи розподілу ресурсів також потребують постійного вдосконалення. Слід вивчати нові методи та технології для покращення ефективності.

Забезпечення відмовостійкості та безпеки в віртуалізованому середовищі є критичним завданням для забезпечення надійності та захищеності вашої IT-інфраструктури.

У результаті досліджень сформульовані певні рекомендації:

У напрямку відмовостійкості:

Створення резервних копій: Регулярно створюйте резервні копії важливих даних і налаштовуйте автоматичне резервне копіювання. Завдяки цьому ви зможете відновити дані у випадку відмови апаратного обладнання або програмних збоїв.

Кластеризація: Використовуйте технології кластеризації для створення відмовостійких конфігурацій. Кластери дозволяють автоматично переключати навантаження на інші сервери у випадку відмови одного з них.

Моніторинг і алерти: Встановіть систему моніторингу, яка відстежуватиме стан обладнання і програмного забезпечення. Налаштуйте автоматичні алерти для оперативного реагування на потенційні проблеми.

Розміщення віртуальних машин на різних серверах: Розподіліть віртуальні машини між різними фізичними серверами, щоб зменшити ризик втрати даних у разі відмови одного сервера.

Тестування відновлення: Регулярно проводьте тести на відновлення для переконанняся, що ви зможете швидко і успішно відновити роботу системи у разі відмови.

У напрямку безпеки:

Патчі і оновлення: Регулярно встановлюйте патчі та оновлення для операційної системи, гіпервізора та програмного забезпечення віртуалізації. Це допоможе усунути відомі вразливості.

Сегментація мережі: Використовуйте правила сегментації мережі для ізоляції віртуальних машин і обмеження доступу до них лише для необхідних користувачів і послуг.

Антивірусне програмне забезпечення: Встановлюйте та оновлюйте антивірусне програмне забезпечення для віртуальних машин для захисту від вірусів і шкідливих програм.

Аудит безпеки: Включіть аудит безпеки, щоб відстежувати події та спроби несанкціонованого доступу до віртуальних машин.

Мінімізація атакованої поверхні: Вимикайте зайві служби та функції, які не використовуються, для зменшення потенційної атакованої поверхні.

Навчання персоналу: Навчіть вашу команду безпеки віртуалізованого середовища, включаючи керування ключами, обробку інцидентів та інші аспекти безпеки.

Моніторинг безпеки: Встановіть систему моніторингу безпеки для виявлення підозрілих активностей та інцидентів безпеки.

Резервне копіювання і відновлення: Забезпечте можливість резервного копіювання і відновлення важливих даних та налаштувань системи у випадку кібератак або інших інцидентів.

Постійне оновлення політик безпеки: Регулярно переглядайте і оновлюйте політики безпеки відповідно до змін у загрозах та потребах вашої організації.

Ізоляція критичних систем: Критичні системи повинні бути додатково ізольовані та мати посилені заходи безпеки.

Захист віртуалізованого середовища вимагає поєднання технічних заходів, політик безпеки та навчання персоналу. Ретельне планування та виконання цих рекомендацій допоможе забезпечити високий рівень відмовостійкості та безпеки.

Отже, розроблений адаптивний метод розподілу ресурсів є потужним інструментом для оптимізації використання обчислювальних ресурсів у віртуалізованих середовищах при виконанні наведених практичних рекомендацій.

4.4. Висновки за розділом 4

Проведено дослідження запропонованого методу адаптивного прогнозування запитів на ресурси на основі тесту на послідовність серій. Проведено порівняння фактичних значень і прогнозованих 5-точкових значень, отриманих за допомогою запропонованого методу прогнозування запитів на ресурси на основі тесту на послідовність серій для попередньо оброблених послідовностей Показано, що використання запропонованого методу прогнозування запитів на ресурси на основі тесту на послідовність серій знижує загальну кількість помилок прогнозу у більшості практичних випадків.

Проведено дослідження запропонованого методу адаптивного розподілу хмарних ресурсів. Для оцінки ефективності розробленого методу були використані такі метрики для порівняння його з методами кругового перебору (RR), SPEA2 та NSGA-II: кількість використовуваних обчислювальних ресурсів, відстань відповідності продуктивності ресурсів, відстань відповідності частки ресурсів, використання ресурсів, динаміка ресурсів, відмовостійкість та забезпечення безпеки. Суттєві переваги були відмічені при порівнянні з методом кругового перебору (RR). Порівняно з методами SPEA2 та NSGA-II, запропонований метод адаптивного розподілу ресурсів потребує менше часу для вирішення багатокритеріальних задач при приблизно невеликому розходженні по інших параметрах. Також розроблений метод отримав суттєву перевагу за критерієм збалансованості (середньоквадратичним відхиленням від математичного сподівання випадкової величини завантаженості віртуальних машин) в середньому на 8%.

Надані практичні рекомендації по використанню методу адаптивного розподілу хмарних ресурсів. Показано, що розроблений метод є потужним інструментом для підвищення ефективності використання хмарних обчислювальних ресурсів у віртуалізованих середовищах, що працюють з використанням моделі «Інфраструктура у якості сервісу», при виконанні наведених практичних рекомендацій.

Основні результати розділу надруковані в наукових працях [4, 7, 8, 15] (додаток А).

ВИСНОВКИ

У дисертації наведено основні результати розв’язання актуальної науково-технічної задачі щодо підвищення ефективності використання хмарних обчислювальних ресурсів при застосуванні технології, що базується на моделі «Інфраструктура у якості сервісу».

Це дало змогу отримати такі науково-технічні і практичні результати:

1) проведений аналіз існуючих методів розподілу ресурсів у хмарних обчислювальних середовищах; зокрема проаналізовані особливості хмарних обчислювальних систем та розподілу хмарних ресурсів; проведений порівняльний аналіз підходів до розподілу ресурсів у хмарному середовищі з різними моделями обслуговування та показано, що найбільші проблеми виникають при використанні моделі обслуговування «Інфраструктура у якості сервісу»; сформульована постановка завдання для розв’язання поставленої задачі;

2) розроблений метод базового виділення ресурсів хмарного середовища користувачу, орієнтованому на модель «Інфраструктура у якості сервісу»; в межах методу запропонований підхід до проведення декомпозиції хмарного середовища на зони, виходячи з визначальних особливостей ресурсів, що надаються в кожній зоні; обґрунтований вибір методу аналізу ієрархій для базового виділення хмарних ресурсів; в результаті отримав подальший розвиток метод базового виділення ресурсів хмарного середовища шляхом попередньої декомпозиції множини доступних ресурсів на зони за допомогою введення нерівномірних шкал та використання методу аналізу ієрархій, що дозволило підвищити рівень балансування завантаження хмарних ресурсів; запропонований метод дозволив підвищити рівень балансування завантаження обчислювальних ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», за показником середнього квадратичного відхилення до 8%;

3) розроблений метод превентивного формування черг запитів на віртуальні машини хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу», який декомпозований на такі послідовні етапи: прогнозування майбутніх запитів на віртуальні машини; визначення необхідності заздалегідь виділяти ресурси; встановлення послідовності запитів на віртуальні машини; даний метод враховує результати аналізу попередніх даних та базується на моделі багатоцільового розподілу хмарних ресурсів, що дозволяє завчасно провести прогнозування завантаженості фізичних пристроїв хмарного середовища та запобігти втратам обчислювального ресурсу; його використання дозволило сформувати можливі черги запитів на найбільш витратний за часом формування хмарний ресурс – віртуальні машини, з відхиленням від реальних запитів не більше 15%; крім того, підхід до адаптивного прогнозування завантаженості хмарних ресурсів на основі тесту на послідовність серій дозволив адаптивно вибирати найбільш відповідний метод прогнозування залежно від характеру даних та їхньої змінюваності, з урахуванням особливостей запитів до ресурсів, за рахунок такої додаткової процедури зменшується час прогнозування до 20%;

4) розроблений метод адаптивного розподілу ресурсів хмарного середовища; розглянута модель багатоцільового розподілу ресурсів, котра використовується як при превентивному формуванні черг запитів на віртуальні машини хмарного середовища, так і при адаптивному розподілі хмарних ресурсів, та послідовно розглядає мінімізацію загальної кількості використовуваних фізичних машин та загальної відстані між ресурсами віртуальних машин та фізичних машин, а також формалізацію та кодування даних в рамках генетичного алгоритму для балансування навантаження ресурсів; метод адаптивного розподілу відрізняється від відомих використанням тестування на послідовність серій, математичного апарату удосконаленого генетичного алгоритму NSGA-II та результатами прогнозу запитів на віртуальні машини, що дозволило підвищити ефективність

використання хмарних обчислювальних ресурсів за рахунок реалізації балансу між ресурсами центрального процесора та оперативної пам'яті та зменшення затримки в обслуговуванні хмарних ресурсів; комплексне використання запропонованих методів дозволило зменшити затримку в обслуговуванні обчислювальних ресурсів в процесі функціонування наданої замовнику віртуальної інфраструктури до 5% за рахунок підвищення рівня балансування завантаження обчислювальних ресурсів хмарного середовища;

5) проведена порівняльна оцінка розроблених та існуючих методів розподілу ресурсів хмарного середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»; зокрема, порівняння фактичних значень і прогнозованих 5-точкових значень, отриманих за допомогою запропонованого методу прогнозування запитів на ресурси показало, що використання цього методу знижує загальну кількість помилок прогнозу у більшості практичних випадків; дослідження запропонованого методу адаптивного розподілу хмарних ресурсів та його порівняння з методами кругового перебору (RR), SPEA2 та NSGA-II довело суттєві переваги при порівнянні з методом кругового перебору, а порівняно з методами SPEA2 та NSGA-II, запропонований метод потребує менше часу для вирішення багатокритеріальних задач при приблизно невеликому розходженні по інших параметрах; також розроблений метод отримав суттєву перевагу за критерієм збалансованості (середньоквадратичним відхиленням від математичного сподівання випадкової величини завантаженості віртуальних машин) в середньому на 8%;

6) проведено впровадження розроблених методів розподілу ресурсів хмарного обчислювального середовища при використанні технології, орієнтованої на модель «Інфраструктура у якості сервісу»; зокрема, надані практичні рекомендації по використанню методу адаптивного розподілу хмарних ресурсів, практичні результати, які отримані, підтверджені актами впровадження та доводять коректність теоретичних положень дисертаційної роботи, високу ефективність розроблених методів; результати роботи

впроваджено в ході дослідницьких робіт у підприємстві ТОВ «Лайнап Тех» (м. Харків) при проектуванні комплексної рекомендаційної системи та у навчальний процес кафедри “Комп’ютерна інженерія та програмування” Національного технічного університету “ХПІ” при викладанні на першому курсу магістратури дисципліни «Оптимізація процесів в мультисервісних системах та мережах»;

Подальші дослідження розподілу ресурсів у хмарному середовищі для підвищення ефективності їх використання та мінімізації витрат при використанні технології, що базується на моделі «Інфраструктура у якості сервісу», пов’язані із вивченням хмарних гіперконвергентних структур.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бабій Ю. О., Нездоровін В. П., Махрова Є. Г., Луцкова Л. П. Хмарні обчислення проти розподілених обчислень: сучасні перспективи. *Вісник Хмельницького національного університету. Сер.: Технічні науки*. 2021. № 6. С. 80-85.

2. Братченко Г. Д., Перелигін Б. В., Банзак О. В. Методи та засоби обробки сигналів. Одеса: Плутон, 2014. 452 с.

3. Волот О. І. Застосування хмарних технологій в обліку та управлінні підприємствами реального сектору економіки. Центральноукраїнський науковий вісник. 2019. Вип. 2(35). С. 190-198. DOI: https://doi.org/10.32515/2663-1636.2019.2(35).190-198

4. Гавриленко С. Ю., Кучук Н. Г., Луйкова-Чуйко Н. В., Собчук В. В. Перерозподіл інформаційних потоків в хмарній системі. *Сучасні інформаційні системи*. 2019. Т. 3, № 2. С. 116-122.

5. Глоба Л., Зщманов С., Суліма С. Метод реконфігурації мережі зв'язку з віртуалізованими ресурсами. *Системи управління, навігації та зв'язку*. Збірник наукових праць. Полтава: ПНТУ, 2019. Т. 1 (53). С. 137-141. DOI: https://doi.org/10.26906/SUNZ.2019.1.137.

6. Годлевський І. М., Туревич М. Д., Медведєв В. В. Інформаційна технологія формування варіантів конфігурації логістичного каналу дистрибуції. *Сучасні інформаційні системи*. 2020. Т. 4, No. 1. С. 63-70. DOI: https://doi.org/10.20998/2522-9052.2020.1.09

7. Давидов, В., Гребенюк Д. Development the resources load variation forecasting method within cloud computing systems. *Сучасні інформаційні системи*. 2020. Т. 4, No. 4. С. 128–135. https://doi.org/10.20998/2522-9052.2020.4.18

8. Досенко А. К. Хмарні технології: прикладні технології сучасних платформ. *Вчені записки ТНУ імені В. І. Вернадського*. 2022. Т. 33 (72), № 1, Ч. 3. С. 257-262. DOI: https://doi.org/10.32838/2710-4656/2022.1-3/41

9. Іванущак Н. М., Пасічник В. В. Моделювання розвитку структур комп'ютерних мереж. *Східно–Європейський журнал передових технологій*. 2013. Т. 3, № 2 (63). С. 13–19.

10. Коваленко А. А., Кучук Г. А. Методи синтезу інформаційної та технічної структур системи управління об'єктом критичного застосування. *Сучасні інформаційні системи*. 2018. Т. 2, № 1. С. 22–27. DOI: https://doi.org/10.20998/2522-9052.2018.1.04.

11. Коваленко А., .Ляшенко О., Даниленко О. Поведінка черг під час використання ієрархічної моделі. *Системи управління, навігації та зв'язку*. Збірник наукових праць. Полтава: ПНТУ, 2019. Т. 2 (54). С. 110-113. DOI: https://doi.org/10.26906/SUNZ.2019.2.110.

12. Крамаренко В.В. Інформаційні системи та структури даних. Дніпропетровськ : Системні технології, 2000. 188 с.

13. Кучук Г. А., Коваленко А. А., Лукова-Чуйко Н. В. Метод мінімізації середньої затримки пакетів у віртуальних з'єднаннях мережі підтримки хмарного сервісу. *Системи управління, навігації та зв'язку*. Полтава . ПНТУ, 2017. Вип. 2(42). С. 117-120.

14. Кучук Н. Г. Мерлак В. Ю., Скороделов В. В. Метод зменшення часу доступу до слабкоструктурованих даних. *Сучасні інформаційні системи* : щоквартальний науково-технічний журнал. 2020. Т. 4, № 1. С. 97-102.

15. Кучук Н. Г. Синтез мережевої моделі комп'ютерної системи на гіперконвергентній платформі. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2020. Вип. 1(59). С. 86-93.

16. Кучук Н. Г. Метод вибору оптимального плану виконання транзакцій e-Learning. *Системи управління та зв'язку*. Полтава : ПНТУ, 2017. Вип. 5(45). С. 83-87.

17. Кучук Н. Г. Метод розрахунку максимальних інтенсивностей інформаційних потоків у гіперконвергентній системі. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2019. Вип. 4(56). С. 53-56.

18. Кучук Н. Г., Нечаусов С. М. Математична модель процесу оперативного перерозподілу обчислювальних ресурсів в гіперконвергентному середовищі. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2017. Вип. 3(43). С. 80-83.

19. Лебеденко Т., Холодкова А. Дослідження методу активного управління чергами на інтерфейсах маршрутизаторів телекомунікаційних мереж. *Системи управління, навігації та зв'язку*. Полтава: ПНТУ, 2019. Т. 4 (56). С. 57-62. DOI: https://doi.org/10.26906/SUNZ.2019.4.057.

20. Литвинова С. Г. Методика проектування та використання хмаро орієнтованого навчального середовища загальноосвітнього навчального закладу. К. : Компринт, 2015. 280 с.

21. Лосєв Ю. І., Руккас К. М. Порівняльний аналіз математичного апарату моделювання телекомунікаційних мереж. *Системи обробки інформації*. Харків : ХУПС, 2007. Вип. 8(66). С. 55-60.

22. Макогон О. А., Мусаєв Р. Г., Дичко О. О. Дослідження процесу обміну інформацією в локальних мережах установ за допомогою використання математичного апарату теорії масового обслуговування. *Сучасні інформаційні системи*. 2019. Т. 3, No. 1. С. 63-70. DOI: https://doi.org/10.20998/2522-9052.2019.1.14

23. Парфьонов Ю. Е. Вибір математичного апарату при розробленні імітаційних моделей інформаційних систем. *Системи обробки інформації*. 2011. Вип. 3 (93). С. 69-72.

24. Петровська І. Ю., Кучук Н. Г., Панченко В. І., Філоненко А. М., Рівномірний розподіл ресурсів, що мають гіперконвергентну інфраструктуру. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2019. Вип. 2(54). С. 119-123.

25. Петровська І. Ю., Коломійцев О. В., Алнаері Фрхат Алі**.** Метод розрахунку розміру буферної пам'яті самовідновлювального сегмента телекомунікаційної мережі. *Системи управління, навігації та зв'язку*. Полтава:

Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2021. Вип. 2 (64). С. 144–147. DOI: 10.26906/SUNZ.2021.2.144.

26. Петровська І. Ю., Кучук Г. А. Розподіл обчислювальних ресурсів у хмарних системах. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2022. Вип. 2 (68). С. 75–78. DOI: 10.26906/SUNZ.2022.2.75.

27. Петровська І. Ю. Методи розподілу ресурсів в хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей сьомої міжнародної науково-технічної конференції (13-15 листопада 2019 р., Черкаси). Черкаси – Баку – Бельсько-Бяла – Харків, 2019. С. 75.

28. Петровська І. Ю., Заполовський М. Й., Шемякін Є. Ю. Система автоматизованного тестування серверної частини мобільного додатку. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали десятої міжн. науково-технічної конференції (9-10 квітня 2020 р., Баку). Баку – Харків – Жиліна, 2020. Т. 2. С.15.

29. Петровська І. Ю., Заполовський М. Й., Мітяєв А. С. Розроблення та дослідження мобільного додатку на основі фреймворку REACT NATIVE. *Проблеми інформатизації* : Тези доповідей восьмої міжнародної науково-технічної конференції (26-27 листопада 2020 р., Черкаси). Черкаси – Баку – Бельсько-Бяла – Харків, 2020. Т. 2. С. 23.

30. Петровська І. Ю., Кучук Г. А., Кучук Н.Г. Підходи до розподілу ресурсів у хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей девятої міжнародної науково-технічної конференції (18-19 листопада 2021 р., Черкаси). Черкаси – Баку – Бельсько-Бяла – Харків, 2021. Т. 2. С. 48

31. Петровська І. Ю., Кучук Г. А. Особливості розподілу ресурсів в хмарних обчислювальних середовищах. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали дванадцятої міжн. науково-технічної конференції (27-28 квітня 2022 р., Баку). Баку – Харків – Жиліна, 2022. Т. 1. С.26.

32. Петровська І. Ю., Кучук Г. А. Порівняння хмарних та туманних обчислень для інтернету речей. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали тринадцятої міжн. науково-технічної конференції (26-27 квітня 2023 р., Баку). Баку – Харків – Жиліна, 2023. Т. 2. С.53.

33. Польщиков К. А., Одарущенко О. Н. Метод оцінки ефективності управління інформаційними потоками в телекомунікаційній мережі спеціального призначення. *Радіоелектронні і комп'ютерні системи* : науково-технічний журнал. 2008. № 6(33). С. 269-276.

34. Роїк О. М., Шиян А. А, Нікіфорова Л. О. Системний аналіз. Вінниця: ВНТУ, 2015. 83 с.

35. Субач І. Ю. Управління розподілом потоків даних в інформаційно-телекомунікаційних мережах. *Наука і техніка Повітряних Сил Збройних Сил України*, 2012. № 3(9). С. 127-129.

36. Субач І., Кучук Н., Чаузов О. Метод рішення задачі розподілу інформаційного ресурсу в АСУ спеціального призначення при варіативному розмірі інформаційних блоків. *Information Technology and Security*. Київ : НТУУ «КПІ», 2016. Т. 4, Вип. 2 (7). С. 269-276.

37. Субач І., Кучук Н., Чаузов О. Моделі розподілу інформаційного ресурсу в АСУ спеціального призначення. *Information Technology and Security*. Київ : НТУУ «КПІ», 2016. Т. 4., Вип. 1 (6). С. 75-83.

38. Суліма С. В. Метод відновлення мережі у віртуалізованому середовищі. *Радіоелектроніка та інформатика*. 2017. № 4(79). С. 4–8.

39. Теоретичні основи формування та деградації складних організаційно-технічних систем : монографія / Є. Б. Смірнов. [та ін.]; ХНУРЕ. Харків: ХНУРЕ, 2018. 162 с.

40. Трубчанінова К. А., Ковтун І. В., Рубльов В. О., Соболєвська Н. В., Дослідження значення величини середньої затримки пакета даних інформаційних потоків у мережах передачі даних. *Інформаційно–керуючі системи на залізничному транспорті*. 2017. № 5. С. 16-25.

41. Чаузов О.М. Моделі розподілу інформаційного ресурсу в АСУ спеціального призначення. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2015. Вип. 4(36). С. 100-102.

42. Afzal S., Kavitha G. Load balancing in cloud computing – A hierarchical taxonomical classification. *J. Cloud Comp.* 2019. Vol. 8, Is. 22. https://doi.org/10.1186/s13677-019-0146-7

43. Aldossary M. A review of dynamic resource management in cloud computing environments. *Computer Systems Science and Engineering*. 2021. Vol. 36, no.3. P. 461–476.

44. Alhihi M. Method of Distribution Network Resources after Restoration, the Networks MPLS-TE Use of Various Telecommunications Technologies to Construct Backbone Networks. *International Journal of Communications, Network and System Sciences*. 2017. Vol. 10. Pp. 251-260.

45. Alibaba Cluster Data: Using 270 GB of Open Source Data to Understand Alibaba Data Centers. URL: https://www.alibabacloud.com/blog/alibaba-cluster-data-using-270-gb-of-open-source-data-to-understand-alibaba-data-centers_594340

46. Alibaba: cluster-trace-gpu-v2023: by Alibaba Group, Workload characterizations. URL: https://github.com/alibaba/clusterdata

47. Amin Salih M., Yuvaraj D., Sivaram M., Porkodi V. Detection And Removal Of Black Hole Attack In Mobile Ad Hoc Networks Using Grp Protocol. *International Journal of Advanced Research in Computer Science*. 2018. Vol. 9, No 6. P. 1–6, DOI: http://dx.doi.org/10.26483/ijarcs.v9i6.6335

48. Attar H., Khosravi M.R., Igorovich S.S., Georgievan K.N., Alhihi M. E-health communication system with multiservice data traffic evaluation based on a G/G/1 analysis method. *Current Signal Transduction Therapy*. 2021. Vol. 16(2). DOI: 10.2174/1574362415666200224094706

49. Ayodeji J. Akande, Colin Fidge, Ernest Foo. Limitations of Passively Mapping Logical Network Topologies. *International Journal of Computer Network and Information Security* (IJCNIS). 2017. Vol. 9, No. 2. P. 1-11. DOI : http://dx.doi.org/10.5815/ijcnis.2017.02.01.

50. Baki A. Continuous monitoring of smart grid devices through multi–protocol label switching. *IEEE Transactions on Smart Grid*. 2014. Vol. 5(3). P. 1210-1215. http://dx.doi.org/10.1109/TSG.2014.2301723.

51. Belgacem Ali. Dynamic resource allocation in cloud computing: analysis and taxonomies. *Computing*. 2022. Vol. 104, is. 3. P .681–710. DOI: https://doi.org/10.1007/s00607-021-01045-2

52. Benson T., Akella A., Maltz D. Unraveling the Complexity of Network Management. *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation* (April 22-24, 2009, Berkeley). Berkeley, CA, USA, 2009. Pp. 335-348.

53. Bogdan P. Mathematical modeling and control of multifractal workloads for data-center-on-a-chip optimization. *Proceedings of the 9th International Symposium on Networks-on-Chip* (September 28-30, 2015, Vancouver). Vancouver, Canada, 2015. Pp. 173-180.

54. Bogdan P., Marculescu R. Non-stationary traffic analysis and its implications on multicore platform design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 2011. No. 30. Pp. 508-519.

55. Bulba S. Composite application distribution methods. *Сучасні інформаційні системи*. 2018. Т. 2, № 3. P. 128–131. DOI: 10.20998/2522-9052.2018.3.22

56. Chan C., Armony M., Bambos N. Maximum weight matching with hysteresis in overloaded queues with setups. *Queueing Systems: Theory and Applications*. 2016. Vol. 82. Pp. 315-351.

57. Chen J., Wang Y. A Hybrid Method for Short-Term Host Utilization Prediction in Cloud Computing. *Journal of Electrical and Computer Engineering*. 2019. P. 1-14. DOI: 10.1155/2019/2782349

58. Chen J., Wang Y. A Resource Demand Prediction Method Based on EEMD in Cloud Computing, *Procedia Computer Science*. 2018. Vol. 131. P. 116-123

59. Chen J., Wang Y., Liu T. A proactive resource allocation method based on adaptive prediction of resource requests in cloud computing. *J Wireless Com Network*. 2021. Vol. 24. DOI: https://doi.org/10.1186/s13638-021-01912-8

60. Ch. Sanjeev Kumar Dash, Ajit Kumar Behera, Satchidananda Dehuri, Ashish Ghosh. An outliers detection and elimination framework in classification task of data mining. *Decision Analytics Journal*. Vol. 6, 2023, DOI: https://doi.org/10.1016/j.dajour.2023.100164

61. W., Tang L., Ji S. Optimizing routing based on congestion control for wireless sensor networks. *Wireless Networks*. 2015. Vol. 22(3). Pp. 1-11.

62. El-Hassany A., Tsankov P., Vanbever L., Vechev M. Network-wide configuration synthesis. Ithaca : Cornell University, 2016. 24 p. URL : http://arxiv.org/abs/1611.02537

63. El-Sherif A., Mohamed A. Joint routing and resource allocation for delay minimization in cognitive radio based mesh networks. *IEEE Transactions on Wireless Communications*. 2014. Vol. 13(1). Pp. 186-197.

64. Fiems D., Dorsman J., Rogiest W. Analysing queueing behaviour in void-avoiding fibre-loop optical buffers. *Performance Evaluation*. 2016. Vol. 103. P. 23-40.

65. Franti P. K-sets and k-swaps algorithms for clustering sets. Pattern Recognition. 2023. Vol. 139, No. 13. 109454. P. 1-29. DOI: 10.1186/s40537-018-0122-y

66. Gelenbe E., Pujolle G. Analysis and synthesis of computer systems. 2nd Edition. *Advances in Computer Science and Engineering*, 2010. Vol. 4. 309 p.

67. George D., Xia C., Squillante M. Exact-Order Asymptotic Analysis for Closed Queueing Networks. *Journal of Applied Probability*. 2012. Vol. 49(2). Pp. 503-520. http://dx.doi.org/10.1239/jap/1339878801.

68. Gomez-Corral A., Garcia M. Maximum queue lengths during a fixed time interval in the M/M/c retrial queue. *Applied Mathematics and Computation*. 2014. Vol. 235. Pp. 124-136.

69. Hamed Dinari. A Survey on Graph Queries Processing: Techniques and Methods. *Int. Journal of Computer Network and Inf. Security* (IJCNIS). 2017. Vol. 9, No. 4. P. 48-56. URL : http://dx.doi.org/10.5815/ijcnis.2017.04.06

70. Honnappa H., Jain R., Ward A. A queueing model with independent arrivals, and its fluid and diffusion limits. *Queueing Systems*. 2015. Vol. 80(1-2). Pp. 71-103.

71. Jiao Z., Zhang B., Gong W., Mouftah H. A virtual queue-based back-pressure scheduling algorithm for wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*. 2015. Vol. 2015(35). Pp. 1-9.

72. John J. Prevost, Kranthi Manoj Nagothu, Brian Kelley, Mo Jamshidi. Prediction of cloud data center networks loads using stochastic and neural models. *2011 6th International Conference on System of Systems Engineering*. 2011. 12138393. DOI: 10.1109/SYSOSE.2011.5966610

73. Kianpisheh S.,A Glitho R. H. Cost-efficient server provisioning for deadline-constrained VNFs Chains: A parallel VNF processing approach. *Proceeding of 2019 16th IEEE Annual Consumer Communications & Networking Conference*. 2019. DOI: 10.1109/CCNC.2019.8651799.

74. Khudov H., Tahyan K., Chepurnyi V., Khizhnyak I., Romanenko K., Nevodnichii A., Yakovenko O. Optimization of joint search and detection of objects in technical surveillance systems. *Сучасні інформаційні системи*. 2020. Т. 4, № 2. P. 156-162. DOI: 10.20998/2522-9052.2020.2.23

75. Kotu V., Deshpande B. Chapter 12 - Time Series Forecasting. *Data Science*. 2019, P. 395-445. DOI: https://doi.org/10.1016/B978-0-12-814761-0.00012-5.

76. Kosenko V. Mathematical model of optimal distribution of applied problems of safety-critical systems over the nodes of the information and telecommunication network. *Сучасні інформаційні системи*. 2017. Т. 1, № 2. С. 4-9. doi:https://doi.org/10. 20998/2522-9052.2017.2.01.

77. Kovalenko A., Kuchuk H., Kuchuk N., Kostolny J. Horizontal scaling method for a hyperconverged network. *International Conference on Information and*

Digital Technologies 2021 (IDT-2021). 2021. P. 331–336. DOI: https://doi.org/10.1109/IDT52577.2021.9497534

78. Kuchuk G., Kovalenko A., Komari I.E., Svyrydov A., Kharchenko V. Improving big data centers energy efficiency: Traffic based model and method. *Studies in Systems, Decision and Control*, vol. 171, Kharchenko, V., Kondratenko, Y., Kacprzyk, J. (Eds.), Springer Nature Switzerland AG. 2019. P. 161-183, DOI: 10.1007/978-3-030-00253-4_8

79. Kuchuk G., Nechausov S., Kharchenko, V. Two-stage optimization of resource allocation for hybrid cloud data store. *International Conference on Information and Digital Technologies*. Zilina, 2015. P. 266-271. DOI: http://dx.doi.org/10.1109/DT.2015.7222982

80. Kuchuk N. H. Comprehensive performance criterion for hyper-converged infrastructure. *Телекомунікаційні та інформаційні технології* : науковий журнал. Київ : ДУТ, 2019. № 3 (64). С. 55-63.

81. Kuchuk N., Bulba S. Mathematical model of distribution of resources between composite applications. *News of science and education*. Sheffield : Science and education LTD, 2017. No. 6 (54). P. 72-80.

82. Kuchuk N., Hani A., Alhihi M., Samour M., Shmatkov S. A Mathematical Model for Managing the Distribution of Information Flows for MPLS-TE Networks under CrC. *Communications and Network*. 2018. Vol. 10, No. 2. P. 31-42.

83. Kuchuk N., Hani Attar, Mohammad R. Khosravi, Shmatkov S. Review and performance evaluation of FIFO, PQ, CQ, FQ, and WFQ algorithms in multimedia wireless sensor networks. *Int. Journal of Distributed Sensor Networks*. 2020. Vol. 16, No. 2. P. 1-9.

84. Kuchuk H., Kovalenko A., Ibrahim B.F., Ruban I. Adaptive compression method for video information. *International Journal of Advanced Trends in Computer Science and Engineering*. 2019. Vol. 8(1). P. 66–69, DOI: http://dx.doi.org/10.30534/ijatcse/2019/1181.22019.

85. Kuchuk N., Mohammed A. S., Shyshatskyi A., Nalapko O. The method of improving the efficiency of routes selection in networks of connection with the

possibility of self-organization. *International Journal of Advanced Trends in Computer Science and Engineering*. 2019. No. 8(1). P. 1-6.

86. Kuchuk N., Mozhaiev O., Semenov S. Brusakova O. Gnusov Y. (2023). Devising a method for balancing the load on a territorially distributed foggy environment. *Eastern-European Journal of Enterprise Technologies*. 2023. Vol. 1(4 (121). DOI: 48–55. https://doi.org/10.15587/1729-4061.2023.274177

87. Kuchuk N., Mukhin V. Kosenko N., Artiukh R. Decomposition Method for Synthesizing the Computer System Architecture. *Advances in Intelligent Systems and Computing*. 2020. Vol 938. Springer, Cham. P. 289-300.

88. Kuchuk N., Shefer O., Cherneva G., Alnaeri F. A. Determining the capacity of the self-healing network segment. *Сучасні інформаційні системи*. 2021. Т. 5, № 2. P. 114–119. DOI: 10.20998/2522-9052.2021.2.16.5.

89. Liang L., Gao D., Leung V. Queue-based congestion detection and multistage rate control in event-driven wireless sensor networks. *Wireless Communications and Mobile Computing*. 2014. Vol. 14(8). Pp. 818-830.

90. Liu X., Zhang D. An Improved SPEA2 Algorithm with Local Search for Multi-Objective Investment Decision-Making. *Appl. Sci.* 2019. Vol. 9, P. 1675. DOI: https://doi.org/10.3390/app9081675

91. Lopes N., Bjorner N., Godefroid P., Jayaraman K., Varghese G. Checking Beliefs in Dynamic Networks. *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation* (May 4-6, 2015, Oakland). Oakland, CA, USA, 2015. Pp. 499-512.

92. Lung-Pin Chen, Fang-Yie Leu, Hsin-Ta Chiao, and Hung-Jr Shiu. Adaptive resource management for spot workers in cloud computing environment. *Int. J. Web Grid Serv*. 2022. Vol. 18, is. 4. P. 437–452. DOI: https://doi.org/10.1504/ijwgs.2022.126124

93. Moqa R, Younas I, Bashir M. Assessing effectiveness of many-objective evolutionary algorithms for selection of tag SNPs. *PLoS One*. 2022. Vol. 17(12). DOI: 10.1371/journal.pone.0278560.

94. Mokhtari Anas, Azizi Mostafa, Gabli,Mohammed. (). Optimizing

management of cloud resources towards best performance for applications execution. *2017 First International Conference on Embedded & Distributed Systems*. 2017. P. 1-5. DOI: 10.1109/EDIS.2017.8284047

95. Nechausov A., Mamusuĉ I., Kuchuk N. Synthesis of the air pollution level control system on the basis of hyperconvergent infrastructures. *Сучасні інформаційні системи.* 2017. Т. 1, № 2. P. 21–26. DOI: 10.20998/2522-9052.2017.2.04

96. Qureshi K. N., Abdullah A. H., Hassan A. N., Sheet D. K., Anwar R. W. Mechanism of Multiprotocol Label Switching for Forwarding Packets & Performance in Virtual Private Network. *Middle–East Journal of Scientific Research*. 2014. Vol. 20, No. 12. P. 2117–2127.

97. Petrovska Inna, Kuchuk Heorhii. Static allocation method in a clod environment with a service model IAAS. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2022. Т. 6, № 3, С. 99–105. DOI: 10.20998/2522-9052.2022.3.13.

98. Petrovska Inna, Kuchuk Heorhii. Adaptive resource allocation method for data processing and security in cloud environment. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2023. Т. 7, № 3, С. 67–73. DOI: 10.20998/2522-9052.2023.3.10.

99. Petrovska Inna, Yefymenko Serhii, Hryhorenko Ihor, Khoroshilo Iurii, Hryhorenko Svitlana. Evaluation of informativeness of indicators in colorimetric control using discriminative analysis. *32 International Scientific Symposium on MMA* (7-11 вересня 2022 р., Созопіль, Болгарія). Созопіль, Болгарія, 2022. DOI: 10.1109/MMA55579.2022.9992712.

100. Petrovska Inna, Kuchuk Heorhii, Mozhaiev Mykhailo. Features of the distribution of computing resources in cloud systems. *2022 IEEE 3rd KhPI Week on Advanced Technology Conference* (3-7 жовтня 2022р., Харків). Харків, 2022. DOI: 10.1109/KhPIWeek57572.2022.9916459.

101. Petrovska I., Kuchuk H. Modeling data processing programs in the self-healing network. *15th international symposium of Croatian metallurgical society SHMD* (22-23 березня 2022 р., Загреб, Хорватія). Загреб, 2022. С. 575.

102. Qiang Ye., Zhuang W. Distributed and adaptive medium access control for internet-of-things-enabled mobile networks. *IEEE Internet of Things Journal*. 2017. Vol. 4, no. 2. P. 446-460. DOI: 10.1109/JIOT.2016.2566659

103. Ruban I., Kuchuk H., Kovalenko A. Redistribution of base stations load in mobile communication networks. *Innovative technologies and scientific solutions for industries*. 2017. No 1 (1). P. 75-81.

104. Saaty T. L. Decision making with the analytic hierarchy process. *Int. J. Services Sciences*. 2008. Vol. 1, No. 1. P. 83-98.

105. Saqib Ul Sabha. A Novel And Efficient Round Robin Algorithm With Intelligent Time Slice And Shortest Remaining Time First. *Materials Today: Proceedings*. 2018. Vol. 5, Issue 5, Part 2. P. 12009-12015. DOI: https://doi.org/10.1016/j.matpr.2018.02.175

106. Semenov S., Cao Weilin. Testing process for penetration into computer systems mathematical model modification. *Сучасні інформаційні системи*. 2020. Т. 4, № 3. P. 133–138. 2020. DOI: 10.20998/2522-9052.2020.3.19

107. Semenov S., Mozhaiev O., Kuchuk N., Kuchuk H. (). Devising a procedure for defining the general criteria of abnormal behavior of a computer system based on the improved criterion of uniformity of input data samples. *Eastern-European Journal of Enterprise Technologies*. 2022. Vol. 6(4(120). P. 40–49. DOI: https://doi.org/10.15587/1729-4061.2022.269128

108. Semenov S., Sira O., Gavrylenko S., Kuchuk N. Identification of the state of an object under conditions of fuzzy input data. *Eastern-European Journal of Enterprise Technologies*. 2019. Vol 1, No 4 (97). P. 22-30. DOI: 10.15587/1729-4061.2019.157085

109. Semenov S., Weilin C., Zhang L., Bulba S. Automated penetration testing method using Deep machine learning technology. *Advanced Information Systems*. 2021. Vol. 5, Issue 3. P. 119–127. DOI: https://doi.org/10.20998/2522-9052.2021.3.16

110. Semenov S., Zhang L., Cao W., Bulba S., Babenko V., Davydov V. Development of a fuzzy GERT-model for investigating common software

vulnerabilities. *Eastern-European Journal of Enterprise Technologies*. 2021. Vol. 6(2(114). P. 6–18. DOI: https://doi.org/10.15587/1729-4061.2021.243715

111. Sen G., Krishnamoorthy M., Rangaraj N., Narayanan V. Exact approaches for static data segment allocation problem in an information network. *Computers & Operations Research*. 2015. Vol. 62. Pp. 282-295. URL: http://dx.doi.org/10.1016/j.cor.2014.05.023.

112. Sivaram M., Yuvaraj D., Amin Salih Mohammed, Porkodi V., Manikandan V. The Real Problem Through a Selection Making an Algorithm that Minimizes the Computational Complexity. *International Journal of Engineering and Advanced Technology*. 2018. Vol. 8, iss. 2. P. 95-100.

113. Taha H. Operations Research. 10th edition. Upper Saddle River : Pearson, 2016. 848 p.

114. Tanenbaum A. Computer Networks. Prentice Hall, 2010. 960 p.

115. Tang F., Tang C., Yang Y., Yang L., Zhou T., Li J., Guo M. Delay-Minimized Routing in Mobile Cognitive Networks for Time-Critical Applications. *IEEE Transactions on Industrial Informatics*. 2017. Vol. 13(3). Pp. 1398-1409.

116. Vaquero L., Rodero-Merino L. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM SIGCOMM Computer Communication Review*. 2014. Vol. 44(5). Pp. 27-32.

117. Vinutha H.P., Poornima B., Sagar, B. Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset. Information and Decision Sciences. 2018. P. 511-518. Doi: 10.1007/978-981-10-7563-6_53.

118. Viviane De Buck, Carlos André Muñoz López, Philippe Nimmegeers, Ihab Hashem, Jan Van Impe. Multi-objective optimisation of chemical processes via improved genetic algorithms: A novel trade-off and termination criterion. *Computer Aided Chemical Engineerin.* Elsevier, 2019. Vol. 46, , P. 613-618. DOI: https://doi.org/10.1016/B978-0-12-818634-3.50103-X.

119. Wu Y., Williamson C. Impacts of data call characteristics on multiservice CDMA system capacity. *Performance Evaluation*. 2015. Vol. 62(1-4). Pp. 83–99. http://dx.doi.org/10.1016/j.peva.2005.07.011.

120. Xiao, S., Li, T., Guo, B., Huang, Z. Retraction Note: Cloud platform wireless sensor network detection system based on data sharing. *Cluster Computing*. 2019., Vol. 22, no. 6. P. 14157-14168. DOI: 10.1007/s10586-018-2260-6

121. Xi N., Sun C., Ma J., Shen Y. Secure service composition with information flow control in service clouds. *Future Generation Computer Systems*. 2015. Vol. 49. Pp. 142-148. http://dx.doi.org/10.1016/j.future.2014.12.009.

122. Yaloveha V., Hlavcheva D., Podorozhniak A., Kuchuk H. Fire hazard research of forest areas based on the use of convolutional and capsule neural networks. *2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering*, UKRCON 2019. DOI: http://dx.doi.org/10.1109/UKRCON.2019.8879867

123. Yu Z., Xu H., Yang Z., Guo B. Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user foot-prints. *IEEE Transactions on Human-Machine Systems*. 2016. Vol. 46(1). Pp. 151-158.

124. Zelentsov, D. Us, S. Koryashkina, L. Stanina O. Solving Continual Two-Stage Problems of Optimal Partition of Sets. *International Journal of Research Studies in Computer Science and Engineering*. 2017. Vol. 4, Is. 4. Pp. 72-80.

125. Zhang S., Malik S., McGeer R. Verification of computer switching networks: An overview. *Proceedings of the International Symposium on Automated Technology for Verification and Analysis* (October 3-6, 2012, Thiruvananthapuram). Thiruvananthapuram, India, 2012. Pp. 1-16.

126. Zhen Xiao, Weija Song, Qu Chen. Dynamic Resource Allocation using Virtual Machines for Cloud Computing Environment. *IEEE transaction on parallel and distributed systems*. 2013. Vol. 24, Is. 6. P. 1107 –1117. DOI: 10.1109/TPDS.2012.283.v.

ДОДАТОК А
СПИСОК НАУКОВИХ ПРАЦЬ

Наукові праці, в яких опубліковано основні наукові результати:

1. Петровська І. Ю., Кучук Н. Г., Панченко В. І., Філоненко А. М. Рівномірний розподіл ресурсів комп'ютерних систем, що мають гіперконвергентну інфраструктуру. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2019, Вип. 2 (54). С. 119–122. DOI: 10.26906/SUNZ.2019.2.119.

2. Петровська І. Ю., Коломійцев О. В., Алнаері Фрхат Алі. Метод розрахунку розміру буферної пам'яті самовідновлювального сегмента телекомунікаційної мережі. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка». 2021, Вип. 2 (64). С. 144–147. DOI: 10.26906/SUNZ.2021.2.144.

3. Petrovska Inna, Kuchuk Heorhii. Static allocation method in a clod environment with a service model IAAS. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2022, Т. 6, № 3. С. 99–105. DOI: 10.20998/2522-9052.2022.3.13.

4. Петровська І. Ю., Кучук Г. А. Розподіл обчислювальних ресурсів у хмарних системах. *Системи управління, навігації та зв'язку*. Полтава: Національний університет «Полтавська політехніка імені Юрія Кондратюка». 2022, Вип. 2 (68). С. 75–78. DOI: 10.26906/SUNZ.2022.2.75.

5. Petrovska Inna, Kuchuk Heorhii. Adaptive resource allocation method for data processing and security in cloud environment. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2023, Т. 7, № 3. С. 67–73. DOI: 10.20998/2522-9052.2023.3.10.

Опубліковані праці апробаційного характеру:

6. Петровська І. Ю. Методи розподілу ресурсів в хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей VII Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2019. С. 75.

7. Петровська І. Ю., Заполовський М. Й., Шемякін Є. Ю. Система автоматизованного тестування серверної частини мобільного додатку. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали X Міжнародної науково-технічної конференції . Баку – Харків – Жиліна. 2020, Т. 2. С.15.

8. Петровська І. Ю., Заполовський М. Й., Мітяєв А. С. Розроблення та дослідження мобільного додатку на основі фреймворку REACT NATIVE. *Проблеми інформатизації* : Тези доповідей IIX Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2020, Т. 2. С. 23.

9. Петровська І. Ю., Кучук Г. А., Кучук Н.Г. Підходи до розподілу ресурсів у хмарних обчислювальних середовищах. *Проблеми інформатизації* : Тези доповідей IX Міжнародної науково-технічної конференції. Черкаси – Баку – Бельсько-Бяла – Харків, 2021, Т. 2. С. 48.

10. Петровська І. Ю., Кучук Г. А. Особливості розподілу ресурсів в хмарних обчислювальних середовищах. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали XII Міжнародної науково-технічної конференції. Баку – Харків – Жиліна, 2022, Т. 1. С.26.

11. Petrovska Inna, Yefymenko Serhii, Hryhorenko Ihor, Khoroshilo Iurii, Hryhorenko Svitlana. Evaluation of informativeness of indicators in colorimetric control using discriminative analysis. *32 International Scientific Symposium on MMA*. Созопіль, Болгарія, 2022. DOI: 10.1109/MMA55579.2022.9992712.

12. Petrovska Inna, Kuchuk Heorhii, Mozhaiev Mykhailo. Features of the distribution of computing resources in cloud systems. *2022 IEEE 3rd KhPI Week on Advanced Technology Conference*. Харків, 2022. DOI: 10.1109/KhPIWeek57572.2022.9916459.

13**.** Kuchuk H., Petrovska I. Modeling data processing programs in the self-healing network. *15th International symposium of Croatian metallurgical society SHMD*. Загреб, Хорватія, 2022. С. 575.

14. Петровська І. Ю., Кучук Г. А. Порівняння хмарних та туманних обчислень для Інтернету Речей. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : Матеріали XIII Міжнародної науково-технічної конференції. Баку – Харків – Жиліна, 2023. Т. 2. С.53.

15. Rezanov B., Semenova A., Petrovska I., Fesenko T. Model for Providing the Second Factor of Authentication Into Authentication Services with Centralized Account Databases. Proceedings of the 5th International Scientific and Technical Conference "Computer and Information Systems and Technologies". Харків, ХНУРЕ, 2021. С. 46-47. URL: http://csitic.nure.ua/article/view/232201.

ДОДАТОК Б
АКТИ ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ НАУКОВИХ ДОСЛІДЖЕНЬ ДИСЕРТАЦІЙНОЇ РОБОТИ

ЗАТВЕРДЖУЮ

Директор ТОВ «Лайнап Тех»
ЄДРПОУ: 45057523
Тарас КІБІТКІН

„20" червня 2023р.

А К Т
впровадження результатів наукових досліджень дисертаційної роботи Петровської Інни Юріївни

Комісія в складі голови – Тараса Кібіткіна і членів: Ольги Підмогильної та Аліни Кібіткіної

склала дійсний акт у тому, що при проектуванні комплексної рекомендаційної системи ТОВ «Лайнап Тех» були використані наступні результати наукових досліджень Петровської Інни Юріївни:

Розроблений метод аналізу ієрархій, який відрізняється від існуючих тим, що дозволяє провести аналіз проблеми; при цьому проблема ухвалення рішення представляється в вигляді ієрархічно упорядкованих компонентів. Запропонований метод дозволяє раціонально використовувати обчислювальні ресурси хмарного середовища, яке використовує модель обслуговування «Інфраструктура як сервіс».

Також, за допомогою методів превентивного розподілу ресурсів, та адаптивного прогнозування на основі тесту на послідовність серій можна виділяти віртуальні ресурси заздалегідь для зменшення затримки в обслуговуванні ресурсів, використовуючи адаптивний підхід для прогнозування майбутніх запитів на ресурси.

Впровадження даних методів дозволяє підвищити оперативність та достовірність прийняття рішень, щодо початкового розподілу ресурсів в комп'ютерних системах при наданні хмарних інфраструктурних послуг, та подальшого розподілу та прогнозування запитів.

УКРАЇНА ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ "Лайнап Тех" код 45057523 м. Харків

Голова комісії Тарас КІБІТКІН

Члени комісії Ольга ПІДМОГИЛЬНА

Аліна КІБІТКІНА

ЗАТВЕРДЖУЮ
Проректор з науково-педагогічної
роботи Національного технічного
університету «ХПІ»
_____Олександр ТРУШ
«23» серпня 2023р

АКТ

Про використання результатів дисертаційної роботи
асистента кафедри «Комп'ютерна інженерія та програмування»
Петровської Інни Юріївни

Матеріал дисертаційної роботи на здобуття наукового ступеня доктора філософії Петровської Інни Юріївни в якій розв'язана задача з розробки методів розподілу ресурсів у хмарному обчислювальному середовищі для підвищення ефективності їх використання та мінімізації витрат при використанні технології, що базується на моделі «Інфраструктура як сервіс» (IaaS) використовується дисертантом у навчальному процесі Національного технічного університету «Харківського політехнічного інституту» на кафедрі «Комп'ютерна інженерія та програмування» при підготовці бакалаврів, магістрів за спеціальністю «Комп'ютерна інженерія», при викладанні навчальної дисципліни «Оптимізація процесів в мультисервісних системах та мережах». В лекційному матеріалі розглядаються теми, проаналізовані в дисертаційному досліджнні: особливості розподілу обчислювальних ресурсів, методи розподілу ресурсів у віртуальних середовищах, порівняльний аналіз підходів до розподілу ресурсів у ХКС з різними моделями обслуговування.

Результати дисертаційної роботи асистента кафедри «Комп'ютерна інженерія та програмування» Національного технічного університету «ХПІ» Петровської І.Ю. відображені у ряді наукових статей, теорія яких використовується студентами першого (бакалаврського) рівня вищої освіти спеціальності «123 Комп'ютерна інженерія» при дипломному проектуванні з тематики розподілу ресурсів .

Завідуючий кафедрою «КІП»
д.т.н. проф. _____Олександр ЗАКОВОРОТНИЙ

Гарант ОНП «Компютерна інженерія»
д.т.н. проф. Сергій ЛЕОНОВ