UDC 004.7 **doi:** https://doi.org/10.20998/2522-9052.2022.3.10

Inna Petrovska, Heorhii Kuchuk

National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine

ADAPTIVE RESOURCE ALLOCATION METHOD FOR DATA PROCESSING AND SECURITY IN CLOUD ENVIRONMENT

Abstract. Subject of research: methods of resource allocation of the cloud environment. **The purpose of the research:** to develop a method of resource allocation that will improve the security of the cloud environment. At the same time, effective data processing should be achieved. **Method characteristics.** The article discusses the method of adaptive resource allocation in cloud environments, focusing on its significance for data processing and enhanced security. A notable feature of the method is the consideration of external influences when calculating the characteristics of cloud resource requests and predicting resource requests based on a time series test. The main idea of this approach lies in the ability to intelligently distribute resources while considering real needs, which has the potential to optimize both productivity and confidentiality protection simultaneously. Integrating adaptive resource allocation methods not only improves data processing efficiency in cloud environments but also strengthens mechanisms against potential cyber threats**. Research results**. To ensure timely resource allocation, the NSGA-II algorithm has been enhanced. This allowed reducing the resolution time of multi-objective optimization tasks by 5%. Additionally, research results demonstrate that effective utilization of various types of resources on a physical machine reduces resource losses by 1.2 times compared to SPEA2 and NSGA-II methods.

Keywords: cloud environment; cloud resources; security; resource allocation; adaptability.

Introduction

The rapid advancement of cloud computing has brought significant changes to the data processing and security landscape, providing unprecedented scalability and flexibility to enterprises and organizations. As cloud environments continue to evolve, efficient resource distribution and utilization become crucial for achieving optimal productivity and reliable security measures. The concept of adaptive resource allocation in clouds emerged as a powerful strategy to address these challenges, enabling dynamic parameter tuning to cater to diverse workloads, data processing requirements, and security concerns.

The rapid growth of cloud computing, coupled with increased data processing volumes and security demands, poses challenges that necessitate novel approaches and strategies. Resource allocation in cloud environments emerges as a key problem, influencing both productivity optimization and data security measures.

Traditional resource allocation methods often fail to consider adjustments based on dynamic system loads, user requirements, cybersecurity risks, and data integrity needs. This results in inefficient resource usage, overloads, and vulnerabilities from a cybersecurity perspective.

Moreover, the complexity arises from the multitude of diverse data processing tasks with varying levels of difficulty, each requiring different resources. Optimal resource allocation needs to encompass this variability and operate in real-time.

Furthermore, ensuring security in cloud computing is a critical issue. With growing data volumes and increasing attack possibilities, safeguarding confidential information processed in cloud environments is imperative. Adaptive resource allocation can impact security levels, necessitating innovative approaches to address these aspects. In this context, there is a need to enhance resource allocation approaches and ensure their adaptability to changing conditions and demands.

Literature Review Article [1] provides an overview of current research in the field of adaptive resource allocation in cloud computing. The authors analyze various approaches to adaptive resource allocation, including dynamic resource management, task scheduling, and quality of service assurance. They also highlight challenges associated with adaptive resource allocation, such as task diversity, speed requirements, and data volume considerations. This underscores the relevance of adaptive resource allocation for data processing and security in the cloud.

In work [2], authors investigate approaches to adaptive resource management in cloud computing. They explore aspects such as load-based resource allocation, load monitoring and forecasting, and resource optimization for efficiency and cost savings. The paper also sheds light on the key challenges of implementing real-time adaptive resource management. Unfortunately, security questions and the impact of cybersecurity risks on resource allocation efficiency are not addressed.

Review [3] examines various approaches to dynamic resource allocation in cloud computing. The authors analyze different allocation algorithms, considering factors like resource utilization efficiency, energy efficiency, and computation costs. They also emphasize the importance of adaptive resource allocation for optimal cloud system functioning. However, similar to the previous review, cybersecurity aspects are not covered.

Article [4] focuses on secure resource allocation in cloud computing. The authors analyze different approaches to ensuring security in adaptive resource allocation, including data encryption, identification, and authentication. Challenges and possible solutions for securing resources in cloud environments are also discussed. However, the article leans toward a more theoretical exploration, lacking practical implementation examples.

Article [5] addresses load redistribution in geographically distributed fog environments to achieve virtual cluster load balance. The necessity and feasibility of developing a universal and scientifically grounded approach to load balancing are highlighted. Nevertheless, both this article and the previous one seem more focused on theoretical foundations rather than practical experimentation.

In summary, these publications showcase various aspects of adaptive resource allocation in cloud computing, including dynamics, security, efficiency, and practical considerations. Analyzing these sources contributes to a deeper understanding of the issue and underscores the relevance of developing an adaptive resource allocation method for data processing and security in the cloud.

Main Part

The strategy of active allocation of cloud resources is aimed at prompt forecasting of future requests for resources and timely adaptation of allocation procedures to dynamic bursts (changes) of requests in the future. This will effectively counter unpredictable or anomalous situations, including cyber-dangerous incidents. A particularly important place in this strategy is occupied by the method of preventive allocation of resources, the structural diagram of which is clearly presented in Fig. 1.

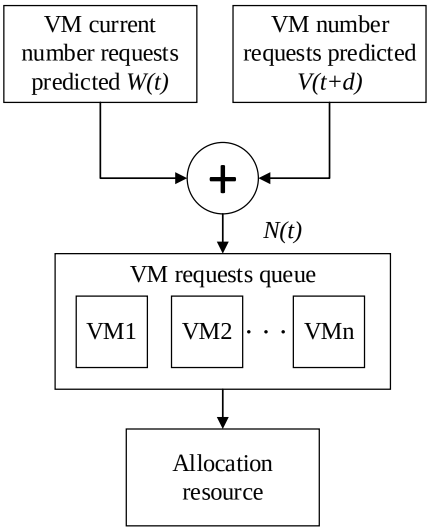

Fig. 1. Structural diagram of the resource's preventive allocation method

The essence of this method is the application of adaptive forecasting procedures based on the analysis of previous data and is based on the parameter R_i (response time). The main focus of the method is aimed at forming a hybrid queue of requests for virtual machines. At the same time, such a queue is formed taking into account current requests, as well as anticipated future dynamic changes.

Suppose that the current sequence of virtual machine requests is denoted by

$$Rec(t)=(rec_1(t),...,rec_i(t),...rec_n(t)), \quad (1)$$

where $rec_i(t)$ represents the number of virtual machines of type i at time t.

To estimate the future number of h main types of requests for virtual machines at the time $t + d$, the adaptive prediction algorithm APMRT is used, which gives the following notation: $V_i(t + d)$ represents the number of requests of the i-th main type for virtual machines at the time $t + h$.

The total number of requests to virtual machines N(t) at time t can be formalized as the sum of the current number of requests Rec(t) and the predicted number of requests $V(t+d)$:

$$V(t+d) = V_1(t+d) + ... + V_i(t+d) + ... + V_h(t+d), \quad (2)$$

where $V_i(t+d)$ is the i-th main type of requests to virtual machines at time $t+d$. The total number of requests for virtual machines $N(t)$ at time t should be equal to the sum of the current number of requests for virtual machines $Rec(t)$ and the predicted number of requests for virtual machines $V(t + d)$:

$$N(t) = W(t) + V(t + d)\times C(t)\times P(t), \quad (3)$$

where $W(t)= rec1(t) + ... + reci(t) + ... + recn(t)$ – is the current number of requests to virtual machines at time t.

If the expected number of requests for virtual machines $V(t+d)$ is not less than the threshold N_{td}, some virtual machines must be allocated resources in advance. $C(t)$ should be equal to 1 and $P(t)$ – is the percentage (for example, 25%) of the virtual machine requests that should be allocated resources in advance given the predicted number of virtual machine requests $V(t+d)$. Otherwise, there is no need to provision virtual machines in advance, i.e. $C(t) = 0$. After determining the predicted number of requests for virtual machines $V(t+d)$, the sequence of requests for virtual machines should be established.

Assume that the predicted number of virtual machine requests is ordered in descending order from the first virtual machine type 1 to h. The largest virtual machines requests (that is, requests of the first type) are at the beginning of the request sequence, and the smallest requests (that is, type h requests) are at the end of the virtual machines request sequence. The intended sequence of requests to virtual machines can be expressed as follows:

$$Rec(t+d)=\begin{pmatrix} rec_1^1(t+d), rec_2^1(t+d), \ldots, \\ rec_j^i(t+d), rec_{j+1}^i(t+d), \ldots, \\ rec_m^h(t+d) \end{pmatrix}, \quad (4)$$

where is $rec_j^i(t+d)$ – the number of requests of the j-th type of virtual machines. Thus, the sequence of requests to virtual machines at time t can be expressed as follows:

$$Rec'(t+d)=\begin{pmatrix} rec_1(t), \ldots, rec_n(t), \\ rec_1^1(t+d), \ldots, rec_j^i(t+d), \\ rec_{j+1}^i(t+d), \ldots, rec_m^h(t+d) \end{pmatrix}, \quad (5)$$

Thus, the method of preventive allocation of resources can be divided into several stages.

Stage 1. Predict future requests for virtual machines:

− using an adaptive forecasting method based on the analysis of previous data and the R_i parameter, determining the predicted number of future main types of requests for virtual machines for a certain time in the future $t+d$;

− designation of the number of requests $Vi(t+d)$ of the i-th main type of virtual machines at the moment of time $t+d$;

− calculation of the total number of requests for virtual machines $N(t)$ at time t by adding the current number of requests $Res(t)$ and the predicted number of requests $V(t+d)$.

Stage 2. Determination of the need to allocate resources in advance:

− calculation of the current number of requests for virtual machines *W(t)* at time *t*;

− if the predicted number of requests for virtual machines *V(t+d)* is not less than the threshold value N_{td}, issuing resources in advance for some virtual machines;

− setting the parameter *C(t)* and the parameter *P(t)*, which should be allocated in advance relative to the predicted quantity *V(t+d)*.

Stage 3. Establishing a sequence of requests to virtual machines:

− taking into account the predicted number of requests for virtual machines *V(t+d)* and establishing the sequence of requests;

− assumption that the predicted number of virtual machine requests is ordered in descending order from the first virtual machine type to h, mapping the largest requests at the beginning of the sequence and the smallest requests at the end;

− finding the predicted sequence of requests to virtual machines and the sequence of requests to virtual machines at time *t*.

Model of multi-objective distribution of resources

The multi-objective resource allocation model is a mathematically and structurally formalized set of algorithms and procedures that focus on task planning and resource load balancing. The model consists of three main parts for the formalization of tasks. The first part creates a stack table containing information about all cloud requests and their execution time on available virtual machines, as shown in Fig. 2.

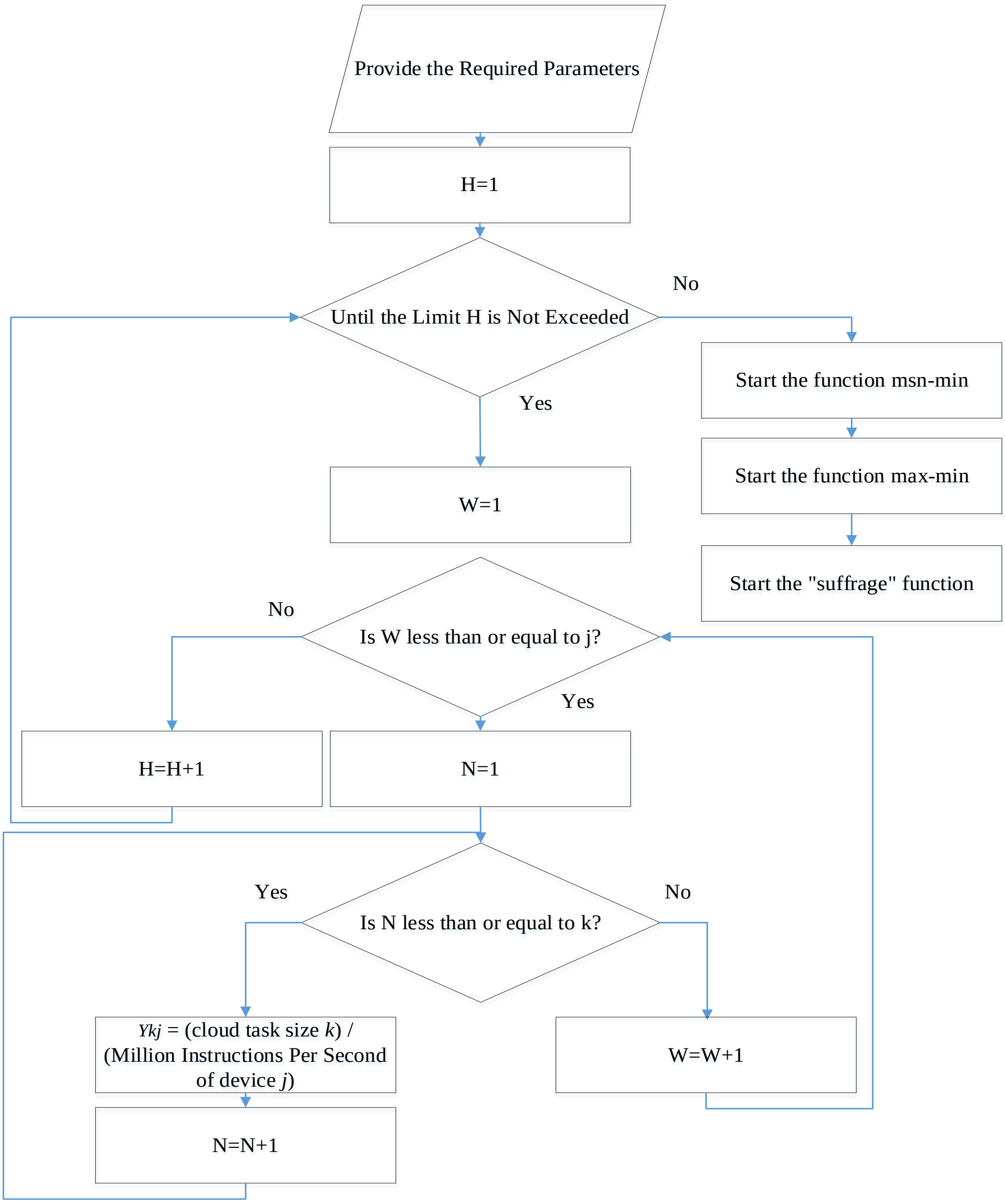

Fig. 2. Block diagram of stack table formation

In the second part, the task of minimizing resources for serving all requests is solved using three scheduling methods: min-min, max-min, and genetic, as shown in Fig. 3, 4 and 5 respectively.

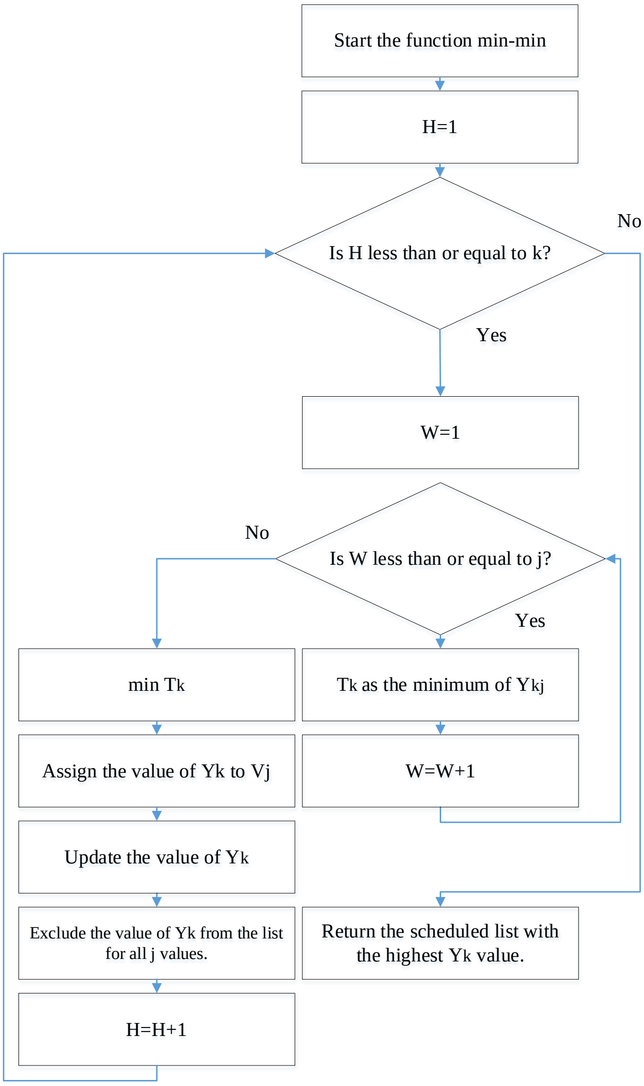

Fig. 3. Block diagram of the resource minimization procedure for serving all requests using the "min-min" method

The work [6–11] presents the various methods of multi-criteria allocation of resources. This method is based on the construction of a multicriteria function with minimization of the number of used physical machines $\min\left(\sum_S x_{i,j}\right)$ and minimizing the overall resource performance mismatch between virtual and physical machines $\min\left(\sum_S WV_{i,j}\right)$ where x_{ij} denotes the mapping element between the virtual machine v_i and the physical machine p_j. If the virtual machine v_i is placed on the physical machine p_j, then x_{ij} is equal to 1. Otherwise, x_{ij} is equal to 0. Therefore, the expression formalizes $\sum_S x_{i,j}$ the total number of physical machines used within the solution S.

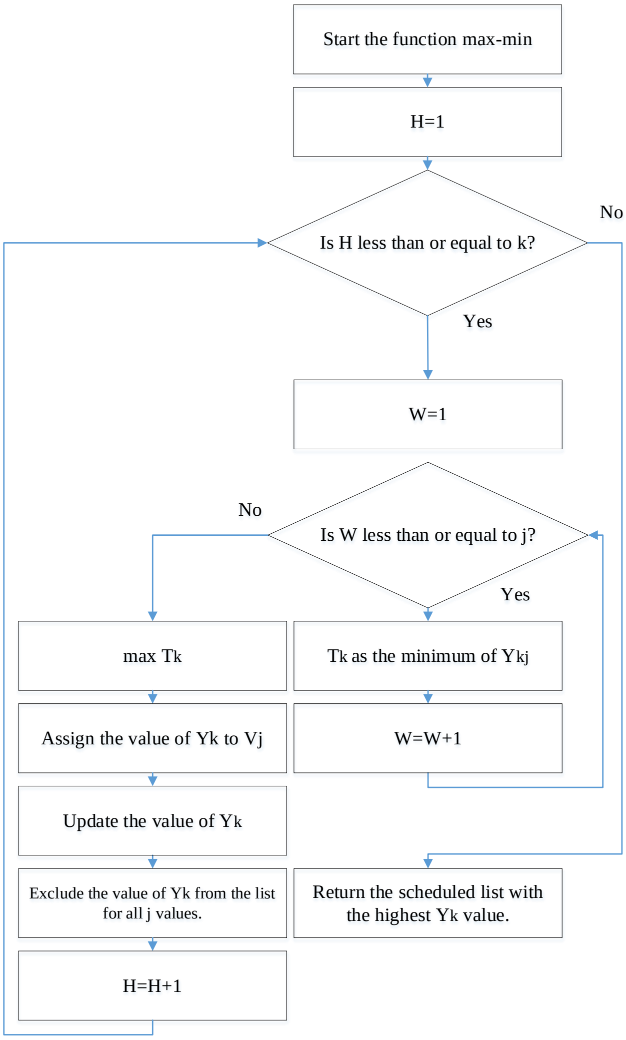

Fig. 4. Block diagram of the resource minimization procedure for serving all requests using the "max-min" method

In the expression to match resource performance

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

a normalized virtual machine performance indicator is used $v_i - y\times v_{i,k}$.

The normalized virtual machine performance indicator vi - and the corresponding physical machine performance normalization indicator $p_j - y\times p_{j,k}$ (y is the normalization factor).

Also, in this expression, $k=1, 2, 3$ formulates the availability of CPU, memory, disk, protection system (firewall) resources, respectively.

To eliminate the shortcomings associated with the inefficient use of physical resources, it is proposed to improve the method of resource allocation based on the prediction of virtual machine requests. If the ratio of different types of resources in the request of the virtual machine is closer to the available resources of the physical machine, that is, the closer the ratio of resources $v_{i1}:v_{i2}:v_{i3}:v_{i4}$ of the virtual machine v_i to $p_{j1}:p_{j2}:p_{j3}:p_{j4}$ of the physical machine p_j is, the less likely it is to lose resources for of this physical machine.

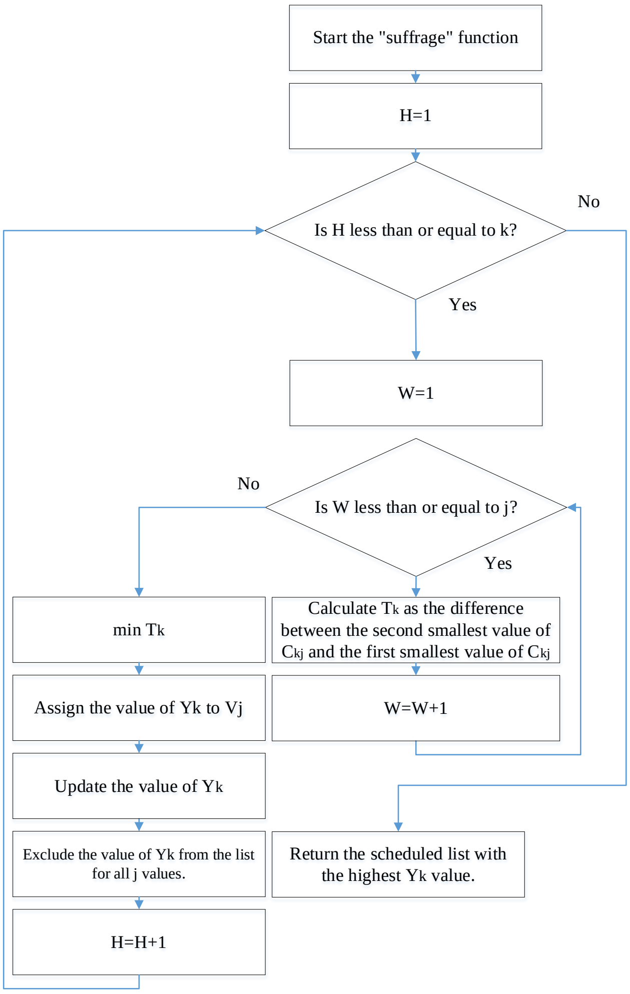

Fig. 5. Block diagram of the resource minimization procedure for serving all requests using the genetic method

Here, v_{i1}, v_{i2}, and v_{i3} represent the requested number of CPU cores, memory, and disk size of virtual machine v_i, respectively; and p_{j1}, p_{j2}, p_{j3}, and p_{j4} denote the available number of CPU cores, memory size, disk size, and firewall resource of physical machine p_j, respectively.

Therefore, a model of matching the proportions of resources is created:

$$PWV_{i,j} = \sqrt{\sum_{k=1}^{4}\left(\left(\frac{y \times p_{j,k} \times v_{i,1}}{p_{j,1}} - y \times v_{i,k}\right)R_k\right)^2}, \quad (6)$$

where p_{jk} – is the available capacity of resource type k for physical machine p_j; v_{ik} – is the requested resource capacity of virtual machine v_i; R_k – is a balancing factor that regulates the value of the complex parameter

$$\left(\frac{y \times p_{j,k} \times v_{i,1}}{p_{j,1}} - y \times v_{i,k}\right) \quad (7)$$

for different types of resources.

For example, if the result of solving the expression (7) for memory resources and firewall resources are 1 and 100, respectively, consideration of firewall resources becomes a more important factor. Therefore, R_k for firewall resources should be set lower than for memory resources, for example, R_k = 1 for memory resources and R_k = 0.1 for firewall resources.

Thus, we formulate the multi-objective problem of optimizing resources taking into account cyber security risks as follows. Allocation based on the number of virtual machines used $\sum_S x_{i,j}$, the total distance between virtual and physical machine resources $\sum_S WV_{i,j}$, and the total distance between resource $\sum_S PWV_{i,j}$ shares requires:

$$\min\left(\sum_S x_{i,j}\right), \quad (8)$$

$$\min\left(\sum_S WV_{i,j}\right), \quad (9)$$

$$\min\left(\sum_S PWV_{i,j}\right). \quad (10)$$

The primary goal of the multi-objective optimization problem (8) for resource allocation is to minimize the total number of physical machines used. This objective depends on the values of the individual mapping elements x_{ij} between the virtual machine v_i and the physical machine p_j within the solution S. The second objective of problem (9) – is to minimize the total distance between the resources of the virtual machines and physical machines within the solution S. This objective depends on the distance between resources $\sum_S WV_{i,j}$ between virtual machine v_i and physical machine p_j. The third goal of problem (10) - is to minimize the total distance between the resource shares of virtual machines and physical machines within the solution S.

This goal is based on the total distance between the resource $PWV_{i,j}$ shares between the virtual machine v_i and the physical machine p_j.

The total resources of processors, memory, disk capacity, and firewall resources requested by the virtual machines hosted on the physical machine p_j are less than the available resources of p_{jdop}. Therefore, the constraints of the optimization problem can be formulated as follows:

$$\sum_S v_{i,j} \times x_{i,j} \le p_{j,1}, \quad (11)$$

$$\sum_S v_{i,2} \times x_{i,j} \le p_{j,2}, \quad (12)$$

$$\sum_S v_{i,3} \times x_{i,j} \le p_{j,3}. \quad (13)$$

$$\sum_S v_{i,4} \times x_{i,j} \le p_{j,4}. \quad (14)$$

The next optimization task is to improve the solution algorithm to speed up the solution speed of the multi-objective optimization function. For this, we will use the classical algorithm for solving the multi-objective optimization problem - NSGA-II [12, 13].

NSGA-II (Nondominated Sorting Genetic Algorithm II) is an evolutionary optimization algorithm used to solve multi-criteria problems. This algorithm belongs to the family of genetic algorithms and is designed to solve problems where there are several conflicting objectives that need to be optimized simultaneously.

NSGA-II is based on the idea of ranking non-dominated solutions (i.e., solutions that cannot be improved in one objective without deterioration in other objectives) and divides the population into Pareto fronts (a set of non-dominated solutions). The main goal of NSGA-II is to find an optimal approximation of the Pareto front, i.e., the set of solutions that best represent the various trade-offs between conflicting objectives.

The main steps of NSGA-II operation include generating an initial population, applying crossover and mutation operators to create new individuals, ranking solutions using non-dominated sorting and criterion ranking, sampling non-dominated solutions for the next population, and applying an archive to store non-dominated solutions and their alternative distributions.

As a genetic algorithm of multi-objective sorting algorithm, this algorithm is widely used to solve multi-objective optimization problems and shows good performance. However, the NSGA-II algorithm has a drawback: the computation time of fitness values (ie, objective functions) is often long, which may threaten the timeliness of resource allocation. In addition, it is necessary to calculate the fitness value for a large number of individuals in the evolution of the population. Thus, we propose to improve the NSGA-II algorithm to speed up the solution process by computing the fitness function in parallel. We use multi-core processors to compute fitness values for individuals in parallel, which accelerates the convergence of the proposed algorithm. The fitness values for each individual are calculated as follows:

$$f_1(E_k) = \sum_S x_{i,j} , \quad (15)$$

$$f_2(E_k) = \sum_S WV_{i,j} , \quad (16)$$

$$f_3(E_k) = \sum_S PWV_{i,j} . \quad (17)$$

Thus, the third part of the modeling is devoted to the formalization and coding of data within the genetic algorithm for resource load balancing.

The block diagram of the algorithm is presented in Fig. 7.

Conducted comparative studies with the SPEA2 [14] and NSGA-II algorithms showed that with 4 threads, the SPEA2 method achieves CPU utilization at the level of 59% and memory at the level of 61%, NSGA-II achieves CPU utilization at the level of 64% and memory at 66%, while the proposed method achieves CPU utilization of 63% and memory utilization of 65%.

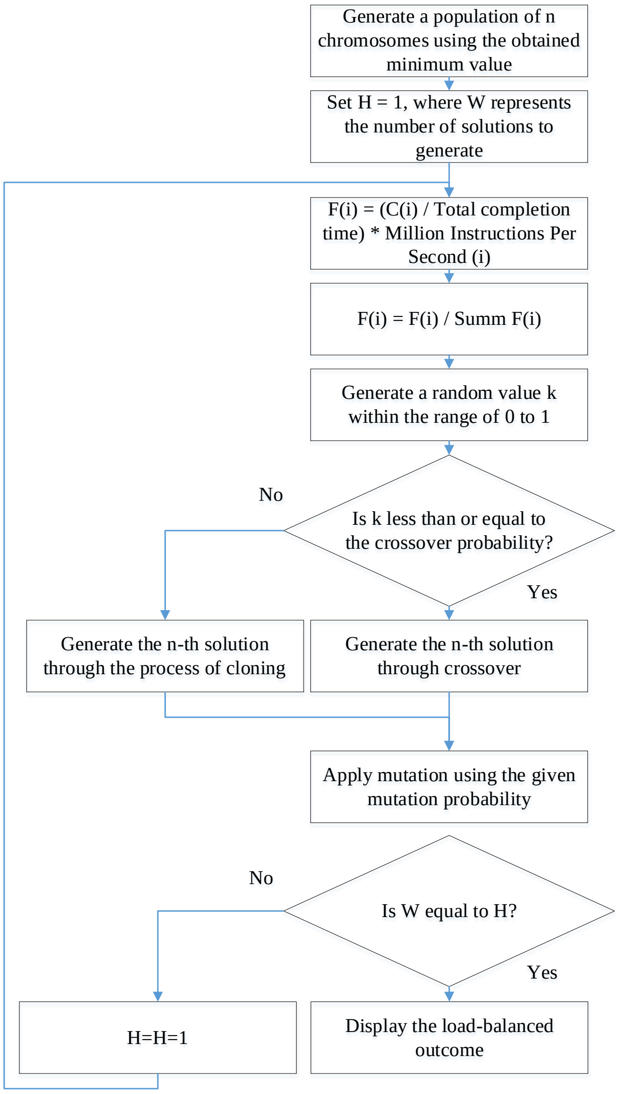

Fig. 7. Block diagram of the genetic algorithm for resource load balancing

Conclusions

Thus, a method of adaptive distribution of cloud resources has been developed.

A distinctive feature of the method is the consideration of external influences when calculating the features of requests for cloud resources and forecasting requests for resources based on the series sequence test.

The NSGA-II algorithm has been improved to ensure timely allocation of resources. This made it possible to reduce the time of solving the multi-objective optimization problem to 5%.

Also, the results of the study showed that the effective use of different types of resources on a physical machine reduces resource losses up to 1.2 times compared to the SPEA2 and NSGA-II methods.

References

1. Mezni, H., Aridhi, S.and Hadjali, A. (2018), "The uncertain cloud: State of the art and research challenges", *International Journal of Approximate Reasoning*, Vol. 103, pp. 139-151, doi: https://doi.org/10.1016/j.ijar.2018.09.009.

2. Nawrocki, P., Grzywacz, M. and Sniezynski, B. (2021), "Adaptive resource planning for cloud-based services using machine learning", *Journal of Parallel and Distributed Computing*, Vol. 152, pp. 88-97, doi: https://doi.org/10.1016/j.jpdc.2021.02.018.
3. Saidi, K., Hioual, O. and Siam, A. (2020), "Resources Allocation in Cloud Computing: A Survey", *ICAIRES 2019: Smart Energy Empowerment in Smart and Resilient Cities*", pp 356–364, doi: https://doi.org/10.1007/978-3-030-37207-1_37.
4. Habiba, U., Masood, R., Shibli, M.A. and Niazi, M.A. (2014), "Cloud identity management security issues & solutions: a taxonomy", *Complex Adapt Syst Model*, Vol. 2, 5, doi: https://doi.org/10.1186/s40294-014-0005-9.
5. Kuchuk, N., Mozhaiev, O., Semenov, S., Haichenko, A., Kuchuk, H., Tiulieniev, S., Mozhaiev, M., Davydov, V., Brusakova, O. and Gnusov, Y. (2023), "Devising a method for balancing the load on a territorially distributed foggy environment", *Eastern-European Journal of Enterprise Technologies*, 1(4 (121), pp. 48–55. doi: https://doi.org/10.15587/1729-4061.2023.274177.
6. Chen, J., Wang, Y. and Liu, T. (2021), "A proactive resource allocation method based on adaptive prediction of resource requests in cloud computing", *J. Wireless Com Network*, 24, doi: https://doi.org/10.1186/s13638-021-01912-8.
7. Petrovska, I. and Kuchuk, H. (2022), "Static allocation method in a cloud environment with a service model IAAS", *Advanced Information Systems*, vol. *6*, *is.* 3, pp. 99–106, doi: https://doi.org/10.20998/2522-9052.2022.3.13.
8. Kuchuk, N., Shefer, O., Cherneva, G. and Alnaeri, F.A. (2021), "Determining the capacity of the self-healing network segment", *Advanced Information Systems*, vol. 5, no. 2, pp. 114–119, Jun. 2021, doi: https://doi.org/10.20998/2522-9052.2021.2.16.
9. Petrovska, I. and Kuchuk H. (2022), "Features of the distribution of computing resources in cloud systems", *Control, Navigation and Communication Systems*, No. 2, pp. 75-78, doi: http://dx.doi.org/10.26906/SUNZ.2022.2.075.
10. Kuchuk, G., Kovalenko, A., Komari, I.E., Svyrydov, A. and Kharchenko, V. (2019), "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, pp. 161-183, doi: https://doi.org/10.1007/978-3-030-00253-4_8.
11. Nechausov, A., Mamusuĉ, I. and Kuchuk, N. (2017), "Synthesis of the air pollution level control system on the basis of hyperconvergent infrastructures", *Advanced Information Systems*, vol. 1, no. 2, , pp. 21–26, doi: https://doi.org/10.20998/2522-9052.2017.2.04.
12. Tan, B., Ma, H. and Mei, Y. (2017), "A NSGA-II-based approach for service resource allocation in cloud", *IEEE Congress on Evolutionary Computation (CEC)*, 17013723, pp. 2574–2581, doi: https://doi.org/10.1109/CEC.2017.7969618.
13. Mohamed, Abdel-Basset, Laila, Abdel-Fatah and Arun Kumar Sangaiah. (2018), "Chapter 10 - Metaheuristic Algorithms: A Comprehensive Review", Editor(s): Arun Kumar Sangaiah, Michael Sheng, Zhiyong Zhang, *Intelligent Data-Centric Systems*, Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications, Academic Press, 2018, Pages 185-231, ISBN 9780128133149, doi: https://doi.org/10.1016/B978-0-12-813314-9.00010-4.
14. Liu, Xi, and Dan Zhang. 2019. "An Improved SPEA2 Algorithm with Local Search for Multi-Objective Investment Decision-Making" *Applied Sciences* 9, no. 8: 1675. doi: https://doi.org/10.3390/app9081675.

Received (Надійшла) 16.05.2023
Accepted for publication (Прийнята до друку) 13.09.2023

Відомості про авторів / About the authors

Петровська Інна Юріївна – аспірантка, кафедра комп'ютерної інженерії та програмування, Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Inna Petrovska – PhD Student of Computer Engineering and Programming Department, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: innap6699@gmail.com; ORCID ID: http://orcid.org/0000-0002-1425-3426.

Кучук Георгій Анатолійович – доктор технічних наук, професор, професор кафедри комп'ютерної інженерії та програмування, Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Heorhii Kuchuk – Doctor of Technical Sciences, Professor, Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: kuchuk56@ukr.net; ORCID ID: http://orcid.org/0000-0002-2862-438X.

Адаптивний метод розподілу ресурсів для обробки даних і підвищення безпеки хмарного середовища

І. Ю. Петровська, Г. А. Кучук

Анотація. Предмет дослідження: методи розподілу ресурсів хмарного середовища. **Мета дослідження**: розробити метод розподілу ресурсів, що дозволить підвищити безпеку хмарного середовища. При цьому повинна бути досягнена ефективна обробка даних. **Характеристики розробки.** У статті розглядається метод адаптивного розподілу ресурсів у хмарних середовищах. Розглянутий метод зосереджується на його значенні для обробки даних та підвищення безпеки. Важливою особливістю методу є врахування зовнішніх впливів при розрахунку характеристик запитів на хмарні ресурси та прогнозування запитів на ресурси на основі перевірки часових рядів. Основна ідея цього підходу полягає в здатності розумно розподіляти ресурси з урахуванням реальних потреб, що має потенціал для оптимізації як продуктивності, так і захисту конфіденційності одночасно. Інтеграція адаптивних методів розподілу ресурсів не тільки підвищує ефективність обробки даних у хмарних середовищах, але й зміцнює механізми проти потенційних кіберзагроз. **Результати дослідження.** Для забезпечення своєчасного розподілу ресурсів алгоритм NSGA-II було вдосконалено. Це дозволило скоротити час розв'язання завдань багатокритеріальної оптимізації на 5%. Крім того, результати дослідження демонструють, що ефективне використання різних типів ресурсів на фізичній машині зменшує втрати ресурсів у 1,2 рази порівняно з методами SPEA2 і NSGA-II.

Ключові слова: хмарне середовище, хмарні ресурси, безпека, розподіл ресурсів; адаптивність.