UDC 004.7 **doi:** https://doi.org/10.20998/2522-9052.2025.1.09

Nina Kuchuk[1], Oleksandr Zakovorotnyi[1], Serhii Pyrozhenko[2], Viacheslav Radchenko[2], Svitlana Kashkevich[3]

[1] National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine
[2] Kharkiv National University of Radio Electronics, Kharkiv, Ukraine
[3] State University "Kyiv Aviation Institute", Kyiv, Ukraine

A METHOD FOR REDISTRIBUTING VIRTUAL MACHINES OF HETEROGENEOUS DATA CENTRES

Abstract. Ant colony algorithms are efficient due to their ability to avoid local minima and find global optima through parallel exploration of multiple solutions. This is achieved through a combination of deterministic and stochastic elements. Deterministic elements include selection rules based on pheromone traces and heuristic information. Stochastic elements introduce an element of randomness and variability. This ensures exploration of new paths. Such a flexible structure makes ant colony algorithms extremely popular in complex and dynamically changing problems, such as the traveling salesman problem and the task of redistributing virtual machines. **The subject of study** in the article is methods of optimizing the placement of virtual machines of heterogeneous data centers. **The purpose of the article** is to improve the efficiency of resource use in heterogeneous virtualized data centers by redistributing virtual machines. **The following results were obtained.** The article presents the developed algorithm for redistributing virtual machines, based on the ant colony metaheuristics, which provides obtaining a migration matrix of virtual machines, differing from known algorithms by taking into account the heterogeneous structure of data centers and additional resources, which represent overhead costs required by the algorithms of virtual machines Live Migration, when calculating it. The developed support system provides flexibility, scalability and a high degree of adaptation to the conditions of modern heterogeneous data centers. It provides the ability to centralize management and monitoring, while supporting the requirements of cross-platform and integration with existing monitoring solutions, which makes it an important tool for ensuring the reliability and efficiency of data centers. **Conclusion.** The redistributing virtual machines support system not only simplifies the tasks of administrators, but also promotes more rational use of computing power, which is especially important in the context of today's dynamic development of IT infrastructure.

Keywords: Virtual Machine; Computer System; ant algorithm; heterogeneity; data center.

Introduction

In this article, a modified ant colony algorithm based on the Ant Colony System metaheuristic is proposed. It is designed to solve the problem of optimizing the placement of virtual machines of heterogeneous data centers. Since ant colony algorithms were initially used to find the shortest path due to their direct purpose, which was to find the shortest path to a food source [1]. For this reason, they are often used to solve the traveling salesman problem, because they have shown their effectiveness in comparison with other algorithms (for example, based on greedy search) [2].

Ant colony algorithms are efficient due to their ability to avoid local minima and find global optima through parallel exploration of multiple solutions [3]. This is achieved through a combination of deterministic and stochastic elements. Deterministic elements include selection rules based on pheromone traces and heuristic information. Stochastic elements introduce an element of randomness and variability. This ensures exploration of new paths. Such a flexible structure makes ant colony algorithms extremely popular in complex and dynamically changing problems, such as the traveling salesman problem [4] and the task of redistributing virtual machines [5, 6].

Therefore, the solution of the traveling salesman problem using the ant colony algorithm was taken as a basis for interpreting this algorithm for solving the problem of redistributing virtual machines of heterogeneous data centers.

The purpose of the article is to improve the efficiency of resource use in heterogeneous virtualized data centers by redistributing virtual machines.

Literature review. The sources [7–9] disclose the development and improvement of modern technologies for processing big data. Which is inextricably linked with the development and improvement of computing infrastructures that support these technologies.

In [10, 11] the complexity of methods and algorithms of such subject areas of big data processing as data mining is presented. Distributed computing is analyzed in [12]. In [13, 14] the recently widely used artificial intelligence systems are considered, which require an appropriate distributed, high-performance, high-load and dynamically scalable computing infrastructure. In [15] the classical ant colony algorithm is described. In [16–18] a distributed system is described in detail. This is a group of autonomous computer systems that are physically separated. But they are connected to a single computer network and are controlled by the software of the distributed system [19–21]. The tasks that independent computers perform interact with each other are also analyzed. The way they share resources and files is considered [22–24].

1. Ant colony algorithm for solving the shortest path problem

The ant colony algorithm was first proposed to solve the shortest path problem [25–27], due to the fact that the shortest path problem itself is the problem of finding the shortest Hamiltonian path in a graph, which is similar to the problem of finding the shortest path to a food source for ants, and also because the traveling salesman problem is one of the most studied NP-hard combinatorial problems [28–30]. The shortest path problem is the problem of finding a minimum-cost closed route among all vertices without repetitions on a complete weighted graph with n

vertices. In the context of the problem, the graph vertices represent cities that need to be visited, and the edge weights indicate distances (lengths) or travel costs.

Like any algorithm based on the use of metaheuristics, it must be modified to solve an applied problem [31]. In this case, it is necessary to modify the ant colony algorithm. It is based on the Ant Colony System metaheuristics, to solve the problem of optimizing the placement of virtual machines of heterogeneous data centers. Since ant colony algorithms were initially used to find the shortest path due to their direct purpose - to find the shortest path to a food source. It is for this reason that they are often used to solve the traveling salesman problem, since they have proven their effectiveness in comparison with other algorithms (for example, based on "greedy" search) [32].

Ant colony algorithms are efficient due to their ability to avoid local minima and find global optima through parallel exploration of multiple solutions. This is achieved through a combination of deterministic and stochastic elements: deterministic elements include selection rules based on pheromone trails and heuristic information, while stochastic elements introduce an element of randomness and variability, allowing exploration of new paths. This flexible structure makes ant colony algorithms extremely useful in complex and dynamically changing problems, such as the traveling salesman problem and the virtual machine reallocation problem. Therefore, the solution of the traveling salesman problem using the ant colony algorithm was taken as a basis for interpreting this algorithm for solving the problem of redistributing virtual machines of heterogeneous data centers.

2. Algorithm for redistributing virtual machines of heterogeneous data centers

This section of the paper presents a generalised scheme of the stages of the algorithm for redistributing virtual machines of heterogeneous data centres based on the ant colony metaheuristic and its description [33, 34]. Using the previous analysis, the scheme of the generalised algorithm for repositioning virtual machines (Fig. 1) can be represented by the sequence of steps to be performed.

Step 1. Initialization of algorithm parameters. At the first stage, the input data is built and the algorithm parameters are initialized. A cluster of physical machines with hosted virtual machines is used as input data, and the scheme for redistributing virtual machines of heterogeneous data centers uses resource allocation models for implementing virtual machines live migration.

Step 2. Formation of homogeneous subsets physical machines. After specifying the input data and initializing the algorithm parameters, the stage of forming homogeneous subsets based on the heterogeneous set is performed physical machines F^g.

Step 3. Search for a local solution. The next step is to create several agents – artificial ants (AA) – and provide each of them with an instance of the input cluster physical machines F^g with distributing virtual machines.

During the execution of the main iteration of the algorithm, the created artificial ants work in parallel and each of them calculates the migration matrix

$$MM_a = \left\langle v, F_i^g \right\rangle, \quad (1)$$

which consists of a list of commands for migrating virtual machines to physical machines for all virtual machines in the cluster.

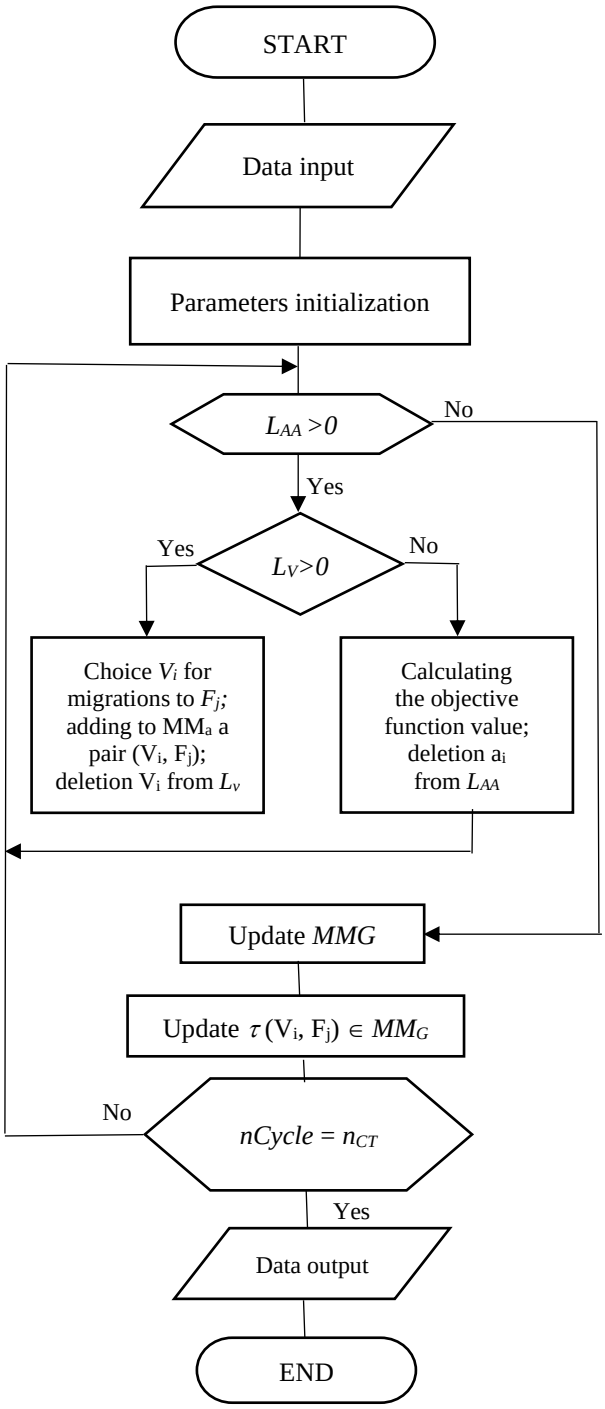

Fig. 1. Block diagram of virtual machine relocation algorithm

In case the source and target physical machines are the same, all migration factors and overheads for these migrated virtual machines will be zero and this will not affect the overall migration costs.

Step 4. Determining the global solution. Then the algorithm determines the best migration matrix MM_G based on the value of the objective function *f*.

Step 5. Global pheromone update. Based on the obtained best migration matrix MM_G, the algorithm updates the global pheromone information and runs the artificial ant again for the next cycle.

Step 6. Terminating the algorithm. Finally, when the specified termination condition is met, the algorithm outputs the best migration matrix MM_G found so far.

3. Evaluation of the complexity of the algorithm for redistributing virtual machines of a virtualized data center

To evaluate the complexity of the virtual machines redistributing algorithm, the Big-O Notation method [35, 36] will be used, which allows for an analysis of changes in the operation of the algorithm in time and algorithmic space with an increase in the volume of input data.

Since with the growth of the amount of input data the number of operations increases and, consequently, the execution time of the algorithm and the amount of memory required for processing the data by the algorithm. This method allows us to estimate the resources required for the algorithm to work when the amount of data changes.

Step 1. Entering initial data **($\mathfrak{I}_1$).** Time complexity depends on the size of the input data, such as: number of iterations (n_{CT}), quantities AA (n_{AA}), quantities virtual machines and physical machines in a cluster ($|V|$ and $|F|$). Time complexity is defined as:

$$\mathrm{O}(n_{CT}) = \mathrm{O}(n_{AA}) = \mathrm{O}(|V|) = \mathrm{O}(|F|) = \mathrm{O}(1);$$

$$\mathrm{O}(\mathfrak{I}_1) = \mathrm{O}(n_{CT}) \cup \mathrm{O}(n_{AA}) \cup \mathrm{O}(|V|) \cup \mathrm{O}(|F|) = \mathrm{O}(1).$$

Step 2. Formation of homogeneous subsets of physical machines **($\mathfrak{I}_2$).** The time complexity depends on the time complexity of the stages of the algorithm for forming homogeneous subsets (the execution time of each of these F^g stages is constant). Time complexity is defined as:

$$\mathrm{O}(k, g) = \mathrm{O}(1); \quad \mathrm{O}(F) = \mathrm{O}(F_g) = \mathrm{O}(n);$$

$$\mathrm{O}(\mathfrak{I}_2) = \mathrm{O}(k, g) \cup \mathrm{O}(F) \cup \mathrm{O}(F_g) = \mathrm{O}(n),$$

where k and g are input parameters; F – physical machines with different types of hypervisors; F^g – physical machines with hypervisors of the same type.

Step 3. Initialize the algorithm parameters **($\mathfrak{I}_3$).** The time complexity of this step is equal to:

$$\mathrm{O}(\mathfrak{I}_3) = \mathrm{O}(v) \cup \mathrm{O}(F_g) = \mathrm{O}(n),$$

where n is number of pairs (v, F^g), because the algorithm simply iterates through each pair (v, F^g) and denotes the value of the pheromone τ_0 equal 0, which requires a constant of time to perform an iteration for each pair.

Step 4. Initialization of the parameters of the artificial ant **($\mathfrak{I}_4$).** The time complexity of the initialization stage of the data structures associated with the artificial ant is:

$$\mathrm{O}(\mathfrak{I}_4) = \mathrm{O}(A_{AA}) = \mathrm{O}(n),$$

where n is number of artificial ants in A_{AA}. This is determined by the fact that the algorithm performs iterations and a constant amount of work for each artificial ant in A_{AA}.

Step 5. Selecting an artificial ant **($\mathfrak{I}_5$).** The time complexity of this step is:

$$\mathrm{O}(\mathfrak{I}_5) = \mathrm{O}(L_{AA}) = \mathrm{O}(n),$$

where n is number of artificial ants in the list L_{AA}, because the algorithm iterates over a set of artificial ants in a list L_{AA}, by selecting an artificial ant at random and performing operations on it until L_{AA} will not become empty.

Step 6. Selecting virtual machines for migration to the physical machines being filled **($\mathfrak{I}_6$).** The time complexity of this stage is:

$$\mathrm{O}(\mathfrak{I}_6) = \mathrm{O}(L_V) = \mathrm{O}(n),$$

where n is number of virtual machines in L_V. This is because the algorithm iterates over each virtual machine in the list once to select a pair (v, F^g) and add it to the migration matrix MM_a, performing operations.

Step 7. Updating the optimal redistributing virtual machines **($\mathfrak{I}_7$).** The time complexity of this step is:

$$\mathrm{O}(\mathfrak{I}_7) = \mathrm{O}(L_{AA}) = \mathrm{O}(n),$$

where n is number of artificial ants in A_{AA}, since the algorithm iterates over each artificial ant в A_{AA} exactly once per time.

Step 8. Pheromone update **($\mathfrak{I}_8$).** The time complexity of the pheromone update step is defined as:

$$\mathrm{O}(\mathfrak{I}_8) = \mathrm{O}(F_g \cdot V) = \mathrm{O}(n \cdot m).$$

This is because there are two nested loops that iterate over all the elements in the sets F^g and V, performing constant time operations within each iteration.

Step 9. Checking the condition for the termination of the algorithm **($\mathfrak{I}_9$).** The time complexity of the stage of checking the condition for the termination of the general iteration of the algorithm depends on the time complexity of the stages of the general iteration of the redistributing virtual machines algorithm and the number of iteration cycles CycleTernmconst.

The time complexity of this stage is defined as:

$$\mathrm{O}(\mathfrak{I}_9) = \bigcup_{i=1}^{8} \mathrm{O}(\mathfrak{I}_i) \times n_{CT} = \mathrm{O}(n_{CT} \cdot n \cdot m). \quad (2)$$

It follows that the worst-case time complexity of the redistributing virtual machines algorithm in heterogeneous data centres can be defined as:

$$\mathrm{O}(\mathfrak{I}) = \mathrm{O}(\mathfrak{I}_9) = \mathrm{O}(n_{CT} \cdot n \cdot m).$$

From expression (2), the time complexity of the algorithm redistributing virtual machines in heterogeneous data centers depends significantly on the number of virtual machines and physical machines in data centers.

As the number increases, the size of the best migration matrix increases MM_G, which leads to an increase in the computational load. At each step of the algorithm, it is necessary to evaluate the current state of the physical machines resources, which in turn requires additional time costs.

4. Research results

To test the result of the theoretical evaluation of the computational complexity of the developed algorithm, it was launched on the constructed simulation model of data centers.

The configuration of the computer used in the experiment includes:

– CPU Intel Core i5-2500 CPU;
– RAM – 32 Гб;
– video adapter Intel HD Graphics;
– operating system MS Windows 10 PRO.

During the experiment, the results of the comparative analysis of the algorithm's running time were summarized in Table 1 and presented in the graph in Fig. 2.

Table 1 – **Estimation of time complexity of the virtual machine redistributing algorithm**

Run number	Input data size, physical machines	Algorithm running time *t*, c
1	10	0,55
2	20	2,56
3	30	5,55
4	40	9,51
5	50	13,8
6	60	17,7
7	70	21,02
8	80	24,59
9	90	27,63
10	100	31,08

The results of the experiment allow us to conclude that the execution time of the redistributing virtual machines algorithm increases linearly with the cluster size.

For a cluster of 100 physical machines, the decision time is about 31.08 seconds, which is quite an acceptable execution time for an autonomous algorithm.

Thus, we can conclude that the proposed redistributing virtual machines algorithm is a fast and feasible method in the context of large-scale data centers.

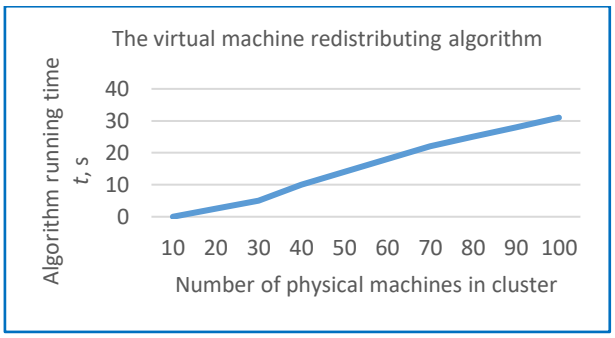

Fig. 2. Dependence of the duration of the redistributing virtual machines algorithm on the increase in the volume of the physical machines cluster

Conclusions

The article presents the developed algorithm for redistributing virtual machines, based on the ant colony metaheuristics, which provides obtaining a migration matrix of virtual machines, differing from known algorithms by taking into account the heterogeneous structure of data centers and additional resources, which represent overhead costs required by the algorithms of virtual machines Live Migration, when calculating it.

The redistributing virtual machines algorithm is presented as a three-procedure algorithm that calculates in parallel the migration matrices of virtual machines for homogeneous subsets of physical machines of data centers.

The developed support system provides flexibility, scalability and a high degree of adaptation to the conditions of modern heterogeneous data centers. It provides the ability to centralize management and monitoring, while supporting the requirements of cross-platform and integration with existing monitoring solutions, which makes it an important tool for ensuring the reliability and efficiency of data centers.

Innovative use of the framework PySyncObj and a competent division of functional zones into two architectural levels allow achieving high performance and reliability in managing virtual machines resources.

The redistributing virtual machines support system not only simplifies the tasks of administrators, but also promotes more rational use of computing power, which is especially important in the context of today's dynamic development of IT infrastructure.

References

1. Subramani, S. and Selvi, M. (2024), “Intrusion detection system and fuzzy ant colony optimization based secured routing in wireless sensor networks”, *Soft Computing*, vol. 28(17-18), pp. 10345–10367, doi: https://doi.org/10.1007/s00500-024-09795-9
2. Ho, J., Park, K. and Kang, D.-K. (2024), “GLNAS: Greedy Layer-wise Network Architecture Search for low cost and fast network generation”, *Pattern Recognition*, vol. 155, number 110730, doi: https://doi.org/10.1016/j.patcog.2024.110730
3. Banupriya, M.R. and Francis Xavier Christopher, D. (2024), “Efficient Load Balancing and Optimal Resource Allocation Using Max-Min Heuristic Approach and Enhanced Ant Colony Optimization Algorithm over Cloud Computing”, *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12(1s), is. 15, pp. 258–270, available at: https://ijisae.org/index.php/IJISAE/article/view/3413
4. Singh, D.R., Singh, M.K. and Chaurasia, S.N. (2024), “An Efficient Hybrid Algorithm with Novel Inver-over Operator and Ant Colony Optimization for Traveling Salesman Problem”, *Communications in Computer and Information Science*, 2093 CCIS, pp. 331–343, doi: https://doi.org/10.1007/978-3-031-64067-4_22
5. Suliman, S.I., Mutalib, A.R.I.A., Yusof, Y.W.M., Rahman, F.Y.A. and Shahbudin, S. (2024), “Energy-Efficient Virtual Machine Placement in Data Centers by Ant Colony Optimization Algorithm (ACO)”, *2024 6th IEEE Symposium on Computers and Informatics*, ISCI 2024, pp. 146–151, doi: https://doi.org/10.1109/ISCI62787.2024.10668096

6. Gomathi, B., Saravana Balaji, B., Krishna Kumar, V., Abouhawwash, M., Aljahdali, S., Masud, M. and Kuchuk, N. (2022), "Multi-Objective Optimization of Energy Aware Virtual Machine Placement in Cloud Data Center", *Intelligent Automation and Soft Computing*, vol. 33(3), pp. 1771–1785, doi: http://dx.doi.org/10.32604/iasc.2022.024052
7. Tong, X. and Wan, Y. (2024), "Information scheduling method of big data platform based on ant colony algorithm", *International Journal of Computer Applications in Technology*, vol. 74(1-2), pp. 1–9, doi: https://doi.org/10.1504/IJCAT.2024.141353
8. Petrovska, I., Kuchuk, H., Kuchuk, N., Mozhaiev, O., Pochebut, M. and Onishchenko, Yu. (2023), "Sequential Series-Based Prediction Model in Adaptive Cloud Resource Allocation for Data Processing and Security", *2023 13th International Conference on Dependable Systems, Services and Technologies, DESSERT 2023*, 13–15 October, Athens, Greece, code 197136, doi: https://doi.org/10.1109/DESSERT61349.2023.10416496
9. Rajkamal, J. and Ezhumalai, P. (2022), "Task scheduling in geo-distributed big data using ant colony optimization", *AIP Conference Proceedings*, vol. 2405, number 030023, doi: https://doi.org/10.1063/5.0074500
10. Ma, J. (2022), "Research on Optimization and Improvement of Intelligent Management System based on Big Data Mining and ant Colony Algorithm", *ACM International Conf. Proc. Series*, pp. 266–271, doi: https://doi.org/10.1145/3558819.3565090
11. Kuchuk, H. and Malokhvii, E. (2024), "Integration of IOT with Cloud, Fog, and Edge Computing: A Review", *Advanced Information Systems*, vol. *8*(2), pp. 65–78, doi: https://doi.org/10.20998/2522-9052.2024.2.08
12. Minarolli, D. (2023), "A Distributed Task Scheduling Approach for Cloud Computing Based on Ant Colony Optimization and Queue Load Information", *Lecture Notes in Networks and Systems*, vol. 571 LNNS, pp. 13–24, doi: https://doi.org/10.1007/978-3-031-19945-5_2
13. Yaqoob, A.A. and Rodeen, W.M. (2024), "Artificial Intelligence Simulation of Ant Colony and Decision Tree in Terms Sustainability", *Lecture Notes in Networks and Systems*, vol. 1033 LNNS, pp. 342–351, doi: https://doi.org/10.1007/978-3-031-63717-9_22
14. Fabre, W., Haroun, K., Lorrain, V., Lepecq, M. and Sicard, G. (2024), "From Near-Sensor to In-Sensor: A State-of-the-Art Review of Embedded AI Vision Systems", *Sensors*, vol. 24(16), 5446, doi: https://doi.org/10.3390/s24165446
15. Tang, Y. and Madina, Z. (2022), "Research on the Flow Space Planning Model of a Classical Garden Based on an Ant Colony Optimization Algorithm", *Journal of Mathematics*, vol. 2022, number 2001084, doi: https://doi.org/10.1155/2022/2001084
16. Zhang, Z. (2023), "A computing allocation strategy for Internet of things' resources based on edge computing", *International Journal of Distributed Sensor Networks*, vol. 17(12), doi: https://doi.org/10.1177/15501477211064800
17. Petrovska, I. and Kuchuk, H. (2023), "Adaptive resource allocation method for data processing and security in cloud environment", *Advanced Information Systems*, vol. 7, no. 3, pp. 67–73, doi: https://doi.org/10.20998/2522-9052.2023.3.10
18. Hunko, M., Tkachov, V., Kuchuk, H. and Kovalenko, A. (2023), Advantages of Fog Computing: A Comparative Analysis with Cloud Computing for Enhanced Edge Computing Capabilities, *2023 IEEE 4th KhPI Week on Advanced Technology, KhPI Week 2023 – Conf. Proc*, 02-06 October 2023, Code 194480, doi: https://doi.org/10.1109/KhPIWeek61412.2023.10312948
19. Li, G., Liu, Y., Wu, J., Lin, D. and Zhao, Sh. (2019), "Methods of Resource Scheduling Based on Optimized Fuzzy Clustering in Fog Computing", *Sensors*, MDPI, vol. 19(9), doi: https://doi.org/10.3390/s19092122
20. Jamil, B. Shojafar, M., Ahmed, I., Ullah, A., Munir, K. and Ijaz, H. (2020), "A job scheduling algorithm for delay and performance optimization in fog computing", *Concurrency and Computation: Practice and Experience*, vol. 32(7), doi: https://doi.org/10.1002/cpe.5581
21. Zhang, Z. (2023), "A computing allocation strategy for Internet of things' resources based on edge computing", *International Journal of Distributed Sensor Networks*, vol. 17(12), doi: https://doi.org/10.1177/15501477211064800
22. Lu, S., Wu, J., Wang, N., Duan, Y., Liu, H., Zhang, J. and Fang, J. (2023), "Resource provisioning in collaborative fog computing for multiple delay-sensitive users", *Software – Practice and Experience*, vol. 53, is. 2, pp. 243–262, doi: https://doi.org/10.1002/spe.3000
23. 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, vol. 1(4 (121), pp. 48–55, doi: https://doi.org/10.15587/1729-4061.2023.274177
24. Kuchuk, G., Nechausov, S. and Kharchenko, V. (2015), "Two-stage optimization of resource allocation for hybrid cloud data store", *Int. Conf. on Information and Digital Techn*, Zilina, pp. 266–271, doi: http://dx.doi.org/10.1109/DT.2015.7222982
25. AlHousrya, O., Bennagi, A., Cotfas, P.A. and Cotfas, D.T. (2024), "A novel Hybrid ant colony algorithm for solving the shortest path problems with mixed fuzzy arc weights", Alexandria Engineering Journal, vol. 109, pp. 841–855, doi: https://doi.org/10.1016/j.aej.2024.09.089
26. Attar, H., Khosravi, M.R., Igorovich, S.S., Georgievan, K.N. and Alhihi, M. (2020), "Review and performance evaluation of FIFO, PQ, CQ, FQ, and WFQ algorithms in multimedia wireless sensor networks", *International Journal of Distributed Sensor Networks*, vol. 16(6), doi: https://doi.org/10.1177/1550147720913233
27. Zhang, J., Xu, X. and Jiang, R. (2024), "The global shortest path planning problem based on the K-means ant colony combination algorithm", *ACM Int. Conf. Proceeding Series*, pp. 247–253, doi: https://doi.org/10.1145/3679409.3679456
28. Gupta, M., Garg, P. and Agarwal, P. (2021), "Ant colony optimization technique in soft computational data research for NP-Hard problems", *Artificial Intelligence for a Sustainable Industry 4.0*, pp. 197–210, doi: https://doi.org/10.1007/978-3-030-77070-9_12
29. Liu, L., Chen, H., Xu, Z. (2022). SPMOO: A Multi-Objective Offloading Algorithm for Dependent Tasks in IoT Cloud-Edge-End Collaboration. Information, 13, 75. doi: https://doi.org/10.3390/info13020075
30. Malik, U.M., Javed, M.A., Frnda, J., Rozhon, J. and Khan, W.U. (2022), "Efficient Matching-Based Parallel Task Offloading in IoT Networks", *Sensors*, vol. 22, doi: https://doi.org/10.3390/s22186906
31. Morfa Hernández, A., Oves García, R., Vázquez Rodríguez, R. and Pérez Risquet, C. (2018), "Integration of visualization techniques to algorithms of optimization of the metaheuristics ant colony", *Computacion y Sistemas*, vol. 22(1), pp. 215–222, doi: https://doi.org/10.13053/CyS-22-1-2769
32. Ali, R., Qaiser, S., El-Kholany, M.M.S., Gebser, M., Leitner, S., Leitner, S., Friedrich, G. (2024), "A Greedy Search Based Ant Colony Optimization Algorithm for Large-Scale Semiconductor Production", *Proc. of the Int. Conference on Simulation and Modeling Methodologies, Technologies and Applications*, pp. 138–149, doi: https://doi.org/10.5220/0012813100003758

33. Kuchuk, H., Kalinin Ye., Dotsenko N., Chumachenko I. and Pakhomov Yu. (2024), "Decomposition of integrated high-density IoT Data Flow", *Advanced Information Systems*, vol. 8, no. 3, pp. 77–84, doi: https://doi.org/10.20998/2522-9052.2024.3.09
34. Zhou, Y. and Wu, W. (2024), "Load-balancing system for SDN data center based on improved ant colony optimization", Proceedings of SPIE - The International Society for Optical Engineering, vol. 13175, doi: https://doi.org/10.1117/12.3031927
35. Kuchuk, N., Kovalenko, A., Ruban, I., Shyshatskyi, A., Zakovorotnyi, O. and Sheviakov, I. (2023), "Traffic Modeling for the Industrial Internet of NanoThings", *2023 IEEE 4th KhPI Week on Advanced Technology*, KhPI Week 2023 - Conference Proceedings, 2023, doi: 194480. http://dx.doi.org/10.1109/KhPIWeek61412.2023.10312856
36. Phalke, S., Vaidya, Y. and Metkar, S. (2022), "Big-O Time Complexity Analysis of Algorithm", *2022 International Conference on Signal and Information Processing*, IConSIP 2022, doi: https://doi.org/10.1109/ICoNSIP49665.2022.10007469

Received (Надійшла) 14.11.2024
Accepted for publication (Прийнята до друку) 13.02.2025

ВІДОМОСТІ ПРО АВТОРІВ/ ABOUT THE AUTHORS

Кучук Ніна Георгіївна – доктор технічних наук, професор, професорка кафедри обчислювальної техніки та програмування, Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Nina Kuchuk – Doctor of Technical Sciences, Professor, Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: nina_kuchuk@ukr.net; ORCID Author ID: http://orcid.org/0000-0002-0784-1465;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57196006131.

Заковоротний Олександр Юрійович – доктор технічних наук, професор, завідувач кафедри комп'ютерної інженерії та програмування, Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Oleksandr Zakovorotnyi – Doctor of Technical Sciences, Professor, Head of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: Oleksandr.Zakovorotnyi@khpi.edu.ua; ORCID ID: http://orcid.org/0000- 0003-4415-838X;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57201613700.

Пироженко Сергій Станіславович – аспірант кафедри електронних обчислювальних машин, Харківський національний університет радіоелектроніки, Харків, Україна;
Serhii Pyrozhenko – Graduate Student of Department of Electronic Computers, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine;
e-mail: serhii.pyrozhenko@nure.ua; ORCID Author ID: http://orcid.org/0009-0006-4209-2144.

Радченко В'ячеслав Олексійович – старший викладач кафедри електронних обчислювальних машин, Харківський національний університет радіоелектроніки, Харків, Україна;
Viacheslav Radchenko – Senior Lecturer of Department of Electronic Computers, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine;
e-mail: viacheslav.radchenko@nure.ua; ORCID Author ID: https://orcid.org/0000-0001-5782-1932;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57189376280.

Кашкевич Світлана Олександрівна – старший викладач кафедри інтелектуальних кібернетичних систем, Державний університет"Київський авіаційний інститут", Київ, Україна;
Svitlana Kashkevich – Senior Lecturer of the Department of Intelligent Cybernetic Systems, State University "Kyiv Aviation Institute", Kyiv, Ukraine;
e-mail: svitlana.kashkevych@npp.nau.edu.ua; ORCID Author ID: https://orcid.org/0000-0002-4448-3839;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=58244269900.

Метод перерозподілу віртуальних машин гетерогенних центрів обробки даних

Н. Г. Кучук, О. Ю. Заковоротний, С. С. Пироженко, В. О. Радченко, С. О. Кашкевич

Анотація. Алгоритми мурашиної колонії ефективні завдяки своїй здатності уникати локальних мінімумів і знаходити глобальні оптимуми через паралельне дослідження багатьох рішень. Це досягається за рахунок комбінації детермінованих і стохастичних елементів: детерміновані елементи включають правила вибору на основі слідів феромонів і евристичної інформації, тоді як стохастичні елементи вводять елемент випадковості і мінливості, забезпечуючи дослідження нових шляхів. Така гнучка структура робить алгоритми мурашиної колонії надзвичайно затребуваними в складних задачах, що динамічно змінюються, як завдання комівояжера і завдання перерозміщення віртуальних машин. **Предметом вивчення** в статті є методи оптимізації розміщення віртуальних машин гетерогенних центрів обробки даних. **Метою статті** є підвищення ефективності використання ресурсів центрів обробки даних за рахунок перерозміщення віртуальних машин. Отримано **такі результати**. У статті представлено розроблений алгоритм перерозміщення віртуальних машин, заснований на метаевристиці мурашиної колонії, що забезпечує отримання матриці міграції віртуальних машин, що відрізняється від відомих алгоритмів з урахуванням її розрахунку гетерогенної структури центру обробки даних і додаткових ресурсів, що становлять накладні витрати, необхідні алгоритмам. Розроблена система підтримки забезпечує гнучкість, масштабованість та високий ступінь адаптації до умов сучасного центру обробки даних. Вона надає можливість централізованого управління та моніторингу, підтримуючи при цьому вимоги кросплатформенності та інтеграції з існуючими рішеннями в галузі моніторингу, що робить її важливим інструментом для забезпечення надійності та ефективності роботи центру обробки даних. **Висновок.** Система підтримки перерозміщення віртуальних машин не лише спрощує завдання адміністраторів, а й сприяє раціональнішому використанню обчислювальних потужностей, що особливо важливо в умовах сучасного динамічного розвитку IT-інфраструктури.

Ключові слова: віртуальна машина; комп'ютерна система; мурашиний алгоритм; гетерогенність; центр обробки даних.