Transaction Planning Methods in Hyperconverged Architecture Systems

Serhii Bulba [1] [0000-0003-0358-7516], Nina Kuchuk [1] [0000-0002-0784-1465], Anna Semenova [1] [0000-0001-7192-5156] and Zhengbing Hu [2] [0000-0002-6140-3351]

[1]National Technical University «KhPI», Kyrpychova str., 2, Kharkiv, Ukraine
nina_kuchuk@ukr.net
[2]Central China Normal University, Wuhan, China
hzb@mail.ccnu.edu.cn

Abstract. The analysis of the features of the functioning of systems with hyperconverged architecture is carried out. Transaction efficiency in such systems is reduced compared to decentralized systems. The purpose of the research: to develop a method for planning transactions in systems with a hypeconvergent architecture, which will take into account the specifics of the functioning of such systems. The development of the method takes into account the centralized management of the transaction package and the distribution of various resources. Existing methods for determining the sequence of transactions in systems with hyperconverged architecture are considered. The methods that were considered were based on the greedy, clustering, and ant algorithms. For each method, its features and functioning scheme are determined. Analysis of existing methods showed the advantages of the greedy algorithm with a small system load. It is also proved that with the growth of information volumes and the number of simultaneously executed transactions, each of the methods considered is less effective than with decentralized management. Therefore, a method for planning the execution of transactions through the sharing of these optimization algorithms is proposed. This allowed to reduce the execution time of the optimal transaction plan in comparison with existing methods. The experiments showed that the effectiveness of the proposed method increases with increasing amount of information. Which is processed by a transaction package of a hyperconverged system.

Keywords: hyperconverged architecture, transaction, greedy, ICT, cluster.

1 Introduction

1.1 Motivation

Today systems with hyperconverged architecture (HCA) are actively developing. Such systems significantly reduce maintenance costs through centralized management. In addition, the extension Software and Hardware becomes cheaper due to the block hardware and software structure. However, systems with hyperconverged archi-

tecture outperform decentralized systems in transaction efficiency. Therefore, new methods of working with transactions that take into account the features HCA. Particularly it is necessary to develop effective methods for planning the execution of transactions in HCA.

1.2 Analysis of related works

System analysis with HCA much attention is being paid. So in [1] hyperconverged platform features analysis conducted. The issues of reducing maintenance costs in systems with HCA are worked out in detail in [2, 3]. The use of HCA in data centers in the analysis and deployment of Big Data is discussed in [4]. Authors in [5] analyzed the possibility of combining HCA with local cloud components. In [6] attention is paid to the features of centralized architectures when working with databases. Data protection methods in HCA systems are addressed in [7, 8].

Transaction planning in various systems in order to reduce the transit time of information flows was considered by many authors, for example [9- 16]. So in [9, 10] considered the use of greedy algorithms. Clustering algorithms are considered in [11-13]. It is shown that with an increase in the volume of information flows, they surpass greedy in efficiency. However, all the considered algorithms do not take into account the specifics HCA. Similar problems arise when applying ant algorithms described in [14-17].

Considering the aforesaid, there is a need to improve the efficiency of transaction planning in systems with HCA. Therefore, new planning methods that take into account the features HCA.

1.3 Goals and structure

One of the important tasks in developing systems with HCA is to increase the efficiency of transaction package execution. The goal of this chapter is to describe appropriate approaches intended for such problem solution. A mechanism for solving this problem is proposed. Initially, various transaction planning methods are analyzed. Then, existing methods for planning the execution of transactions in systems with HCA are considered. Based on their analysis, a more effective, combined method is proposed.

The material is structured as follows. Sections 2 and 3 describe methods based on the greedy and clustering algorithm. Section 4 describes the ant algorithm and proposes a combined method. Section 5 is devoted to the results of a study of the effectiveness of the proposed method.

2 Planning the PCS transaction package using greedy algorithms

Greedy algorithms are most effective when planning without taking into account deadlines for completing transactions. The greedy algorithm makes the best decision

based on the data currently available. The essence of the greedy algorithm lies in the locally optimal choice at each step. To implement the algorithm, you must define the following elements of the problem:

a set of options from which to choose;

a selection function with which the best option is found;

a usability function that determines the usability of the obtained set;

a purpose function that assesses the value of the solution and is not usually explicitly expressed;

A decoupling function that indicates that a final solution has been found. The main disadvantage of greedy algorithms – high probability that solutions found will not be optimal. The main advantage is the speed of finding a solution. It was therefore advisable to use them in finding a rational solution.

If you use greedy algorithms to plan for the use of hyperconversion resources, you should [18]:

1) consider a set of tasks at each stage of resource allocation, which may belong to different transactions;

2) an existing set can contain only tasks, which can be executed at the moment; this is the criterion for choosing a task (CCT);

3) select one task from a set (by task selection criteria), after that, the selected task will receive the resource according to the second resource assignment criterion (CRA, Fig. 1).

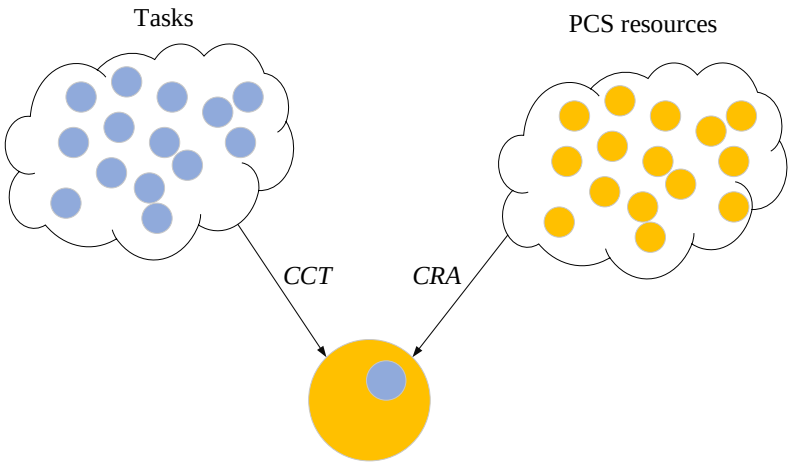

Fig. 1. Planning tasks with two criteria

In this case, the resource assignment decision is made before the tasks are started. A static plan is generated.

Note the following:

1) to select a task, you need to use criteria based on taking into account the deadline for completing the task (and not the deadline);

2) the resource assignment criterion allows you to determine the type of resource for the task. The task completion time is calculated taking into account the current resource load.

The principle of choosing set tasks *StPar*, intended for distribution is such:

1) the set *StPar* formed from tasks that have no outstanding predecessors;

2) completed tasks are removed from the set *StPar*.

To replenish the set *StPar* there are following methods:

1) the method *PlanT*: condition tasks added to the set *StPar* only if the conditions are met:

a) the task is not distributed and is not contained in the set;

b) all predecessors have been distributed for the task;

2) the method *ExecutT*: the set *StPar* is formed by adding tasks in which all parent tasks have been completed.

The developed distribution method has the following form:*GreedPlan* (CCT, CRA, *PlanT*|*ExecutT*).

A large flow chart using the GreedPlan method using greedy is shown in Fig. 2.

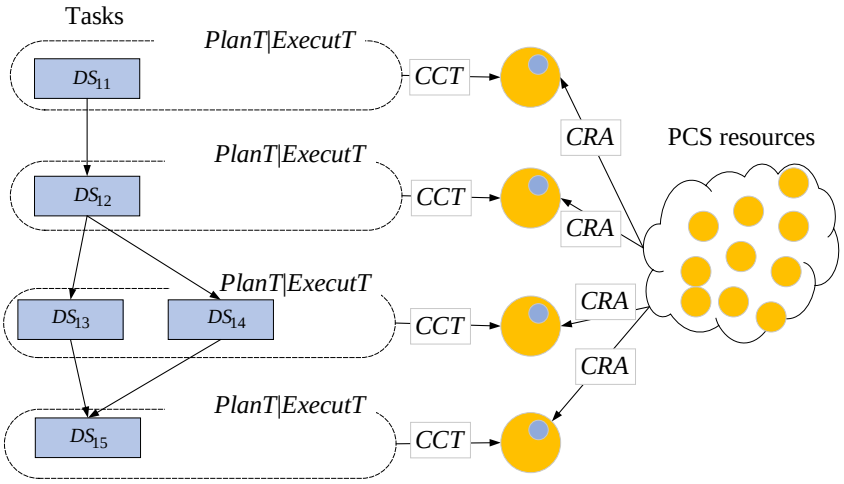

Fig. 2. Large scheme of the method *GreedPlan*

3 Planning of HCS transaction package execution using task clustering methods

When distributed based on task clustering, many tasks are divided into subsets depending on resource requirements. With vertical clustering, each transaction is divided into groups of tasks, taking into account the end of previous tasks (Fig. 3). With horizontal clustering, it becomes possible to enter different tasks into the same cluster. In horizontal clustering, accounting for the end of previous tasks is also necessary. An example is given on Fig. 4 [19].

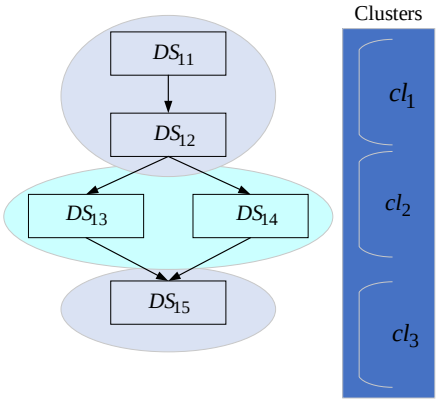

Fig. 3. Vertical clustering

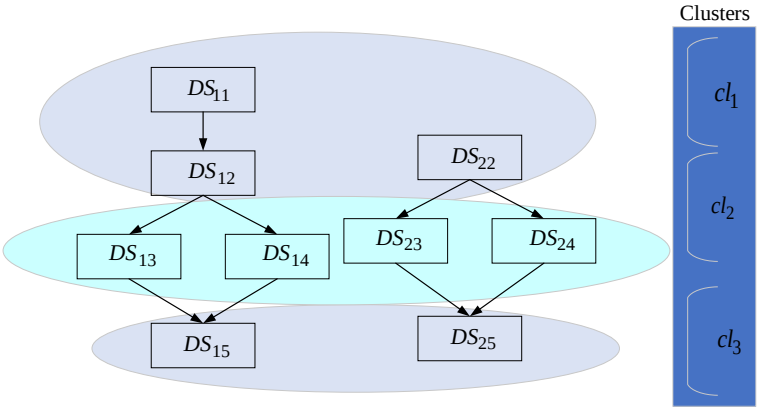

Fig. 4. Horizontal clustering

The task distribution process using the presented clustering methods is divided into stages. Each stage is assigned a set of tasks to be distributed. Adding a task to the set of *StPar* tasks is done only using the *ExecutT* method. The clustering based distribution step is to place a subset of the tasks of one or more transactions. The number of stages depends on the way tasks are grouped. Also, when distributing tasks into clusters, it is necessary to take into account the sequence of tasks in each transaction.

During clustering set clusters are created. It can be characterized by such parameters as the number of clusters and the cluster parameter. The cluster parameters can be the minimum or maximum values of the following values: the time of completion or start of the task, when calculating the task, and others. During clustering, these parameters may not be known. If during the distribution of the task the clusterization parameters are unknown, then these types of clusters arise:

1) consolidation of all transaction tasks in one cluster;
2) task based clustering.

The first approach does not allow even a minimal influence on the distribution process. In this case, there is no way to influence the beginning or completion of a particular transaction. With this approach, the number of clusters is the same as the number of transactions. Such a distribution is always correct. It makes it possible to take into account the priority of transactions. But this approach does not allow for a fair distribution of resources when executing a transaction package.

Consider a few cases.

1. The number of clusters and clustering parameters specified. In this case, static clustering occurs. With such clustering, it is necessary to take into account the correct distribution after the clustering process.

2. The number of clusters is specified. Clustering parameters are assign during clustering. In this case, each task is assigned to a specific cluster depending on the distribution criterion. For the correct distribution of tasks in the cluster, it becomes necessary to monitor the correctness of the distribution. Therefore, it is necessary to consider tasks depending on their predecessors or using methods *PlanT* or *ExecutT*.

3. The number of clusters changes dynamically, and the clustering parameters are set during clustering. There are such clustering ways:

a) federated clustering: – each first tasks distributed in a separate cluster. After that, the parameters for clustering and the criteria for the end of clustering are assign;

b) dynamic clustering – one stage of clustering in this case includes the selection of the task for which the cluster will be determined; assigning tasks to an existing cluster or creating a new cluster depends on the cluster parameter.

After choosing the method of clustering, there is a need for a means of finding the correct order of distribution of clusters on the resources of the hyperconverged system. You can use the *PlanT* or *ExecutT* methods. In this case, clusters are ordered once before the distribution of resources. The search for the next cluster is carried out before each stage of resource allocation. Thus, the number and method of mutual arrangement of the three main stages of distribution using the clustering approach can vary. This is influenced by a specific algorithm [20].

So, we can offer a method for planning the execution of transaction packages of a hyperconverged system based on a clustering approach. This method is *ClustPlan.* Imagine a large flowchart using the method *ClustPlan* on Fig. 5.

4 Planning the execution of a HCS transaction package using an ant algorithm

4.1 Direct use of the ant algorithm

Ant algorithms can be used to solve not only static, but also dynamic problems. The basis of the behavior of ants is self-organization. This is a large number of dynamic mechanisms. These mechanisms ensure that the system achieves a global goal as a result of the low-level interaction of its elements. The principal feature of this interaction is the use by the elements of the system of only local information. This excludes any centralized management. Self-organization is the result of the

interaction of four components: randomness, multiplicity, positive feedback, negative feedback.

In order to have an idea of the operation of the ant algorithm, we present its simplified version on Fig. 6.

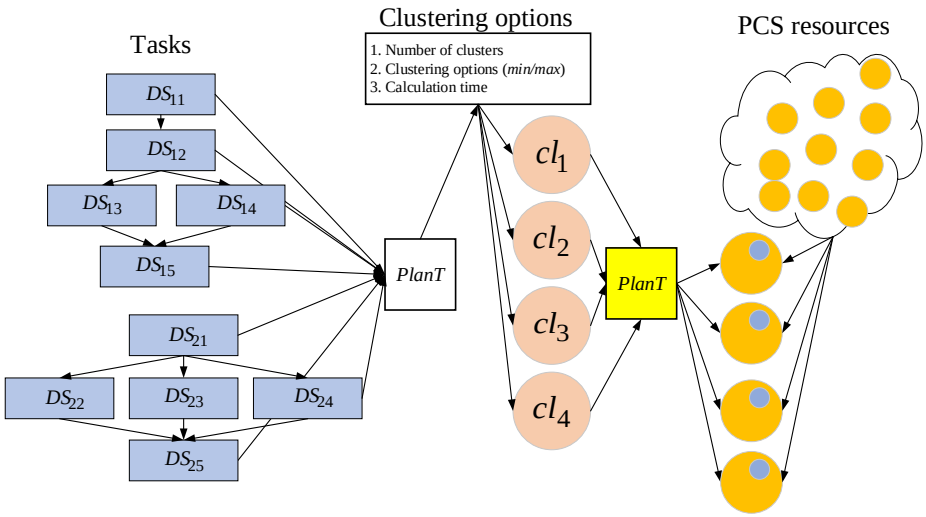

Fig. 5. Large scheme of the method *ClustPlan* functioning

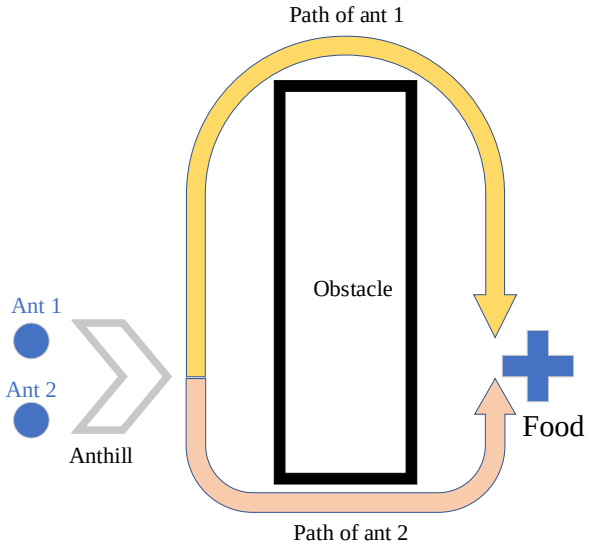

Fig. 6. Simplified scheme of ant algorithm

The main stages of the ant algorithm work include:

1) representation of the problem in the form of a set of components and transitions on which ants can build solutions;

2) determination of the value of heuristic metrics "pheromones;

3) determination of the heuristic of ant behavior when constructing a solution;

4) the implementation of effective local search;

5) the choice is necessary algorithm;

6) parameter settings of the ant colony algorithm.

The determining parameters in the construction of the algorithm are:

1) the number of ants;

2) the balance between study and use;

3) combination with greedy heuristics or local search;

4) the moment when the pheromone is updated.

The efficiency of ant algorithms grows with the growth of the dimension of computational problems. Therefore, with a large number of simultaneously executed transactions, such algorithms will be more efficient than previously considered ones.

In our case, we consider ants to be an existing transaction task. By counting resources of a certain type on which a task can be calculated. Pheromone is a block of statistics. It contains the values of the criteria for completing a task. The performance criteria include the minimum or maximum time to complete the task, resource loads, values of the criteria for the efficiency of resource use, communication time between resources.

At a certain time interval, a method (greedy ant) is shown. It allows you to find the best option for allocating tasks to resources with a specific group of criteria. This approach allows you to implement a resource search. It will be best for the distribution of tasks depending on the selected criterion.The general scheme of the distribution using the ant algorithm is shown in Fig. 7.

4.2 Cluster distribution with the introduction of the ant algorithm

When allocating HCS resources using clustering, it becomes possible to improve the method of constructing a distribution plan using additional conditions. These conditions include the resource allocation method based on the ant algorithm. This will allow more flexibility expanding the capabilities of the clustering approach. When upgrading, a complication of the clustering algorithm occurs. This leads to an increase in the number of iterations when finding the optimal plan. So the time to build it will increase. But this approach can facilitate the operation of the ant algorithm.

In this case, a certain cluster is considered an ant. Which was formed using the clustering parameter. The method of distribution of resources based on the ant algorithm can be used with both vertical and horizontal clustering after their complete improvement. But for each type of clustering, it is necessary to use different initial parameters of attractiveness. The weathering parameters of statistical attractiveness need to be used depending on the distribution method. The general scheme of the ant algorithm based on the clustering approach is shown in Fig. 8.

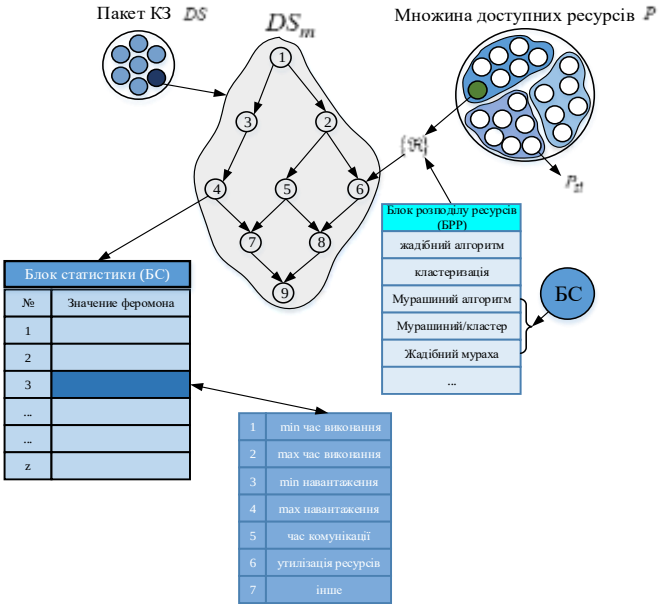

Fig. 7. Using the “greedy ant” method

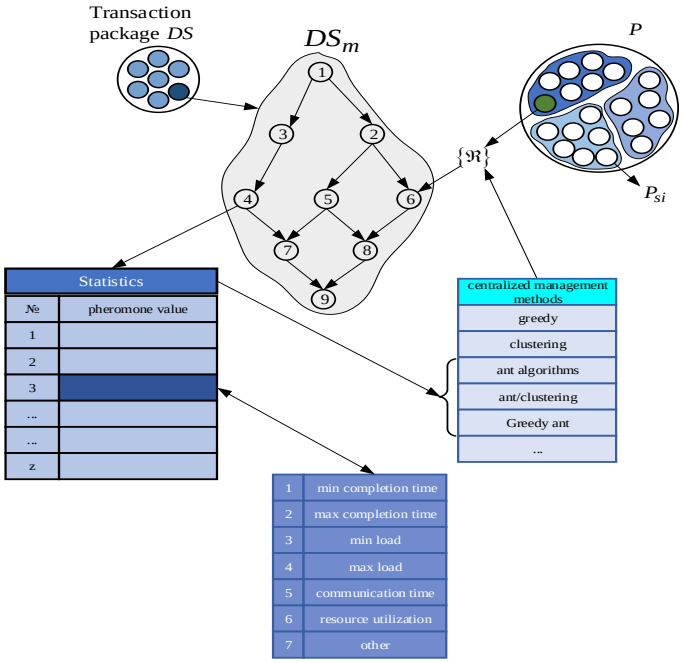

Fig. 8. Allocation of resources by ant algorithm based on clustering approach

The sequence of actions when using the cluster allocation method using the ant algorithm is shown in Fig. 9.

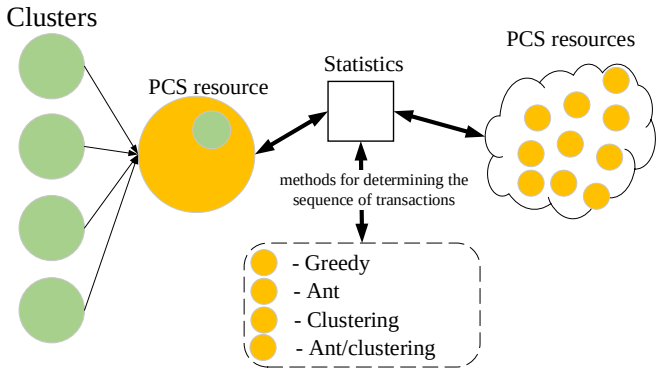

Fig. 9. Scheme of cluster distribution method using ant algorithm

5 Discussion of results

Experiments for small amounts of information. They showed that the execution speed of a HCS transaction package using a greedy algorithm is 22%, and that of an ant is almost 4% longer than a cluster one. This is facilitated by the ability to quickly group the tasks of a transaction package by the parameters considered. The ant algorithm has lost its effectiveness due to the fact that each task is transmitted with a large delay through communication. Complex statistics are also needed.

But the execution speed of a HCS transaction package with a large amount of data using the cluster-ant algorithm is dominated by the greedy algorithm by almost 25%, the ant algorithm by 13%, and the cluster algorithm by 11%. This is due to a decrease in the complexity of the ant algorithm through the distribution of clusters rather than tasks. That is, the time to transfer large amounts of data has been reduced. The results of this experiment is shown in Fig. 10.

6 Conclusions

The analysis of the features of the functioning of systems with hyperconverged architecture is carried out. The methods that were considered were based on the greedy, clustering, and ant algorithms. Therefore, a method for planning the execution of transactions through the sharing of these optimization algorithms is proposed. This allowed to reduce the execution time of the optimal transaction plan in comparison with existing methods. The experiments showed that the effectiveness of the proposed method increases with increasing amount of information. Which is processed by a transaction package of a hyperconverged system.

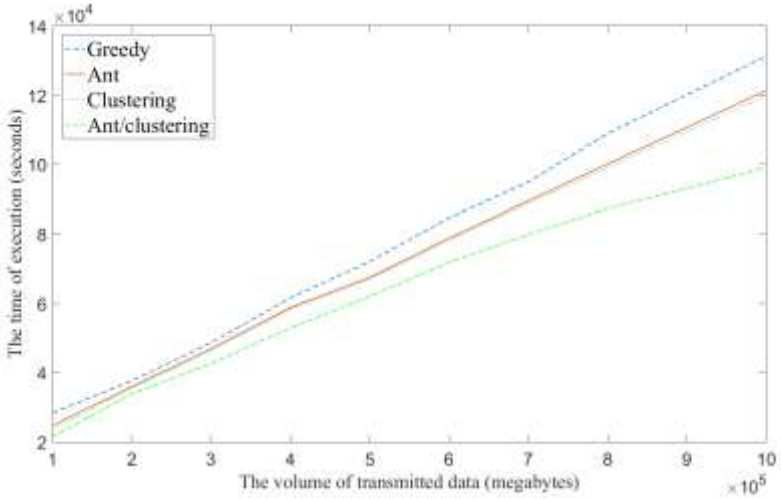

Fig. 10. HCS transaction execution time dependence on the distribution algorithm

References

1. White Paper: Riverbed Hyper-converged Edge, https://www.riverbed.com/document-repository/white-paper--riverbed-hyper-converged-edge.html, last accessed 2019/05/24.
2. Merlac, V., Merlac, V., Smatkov, S., Kuchuk, N., Nechausov, A.: Resourses Distribution Method of University e-learning on the Hypercovergent platform. In: Conf. Proc. of 2018 IEEE 9th Int. Conf. on Dependable Systems, Service and Technologies, DESSERT'2018, Kyiv, May 24-27, pp. 136-140, doi: http://dx.doi.org/ 10.1109/DESSERT.2018.8409114 (2018).
3. Kuchuk, N., Mozhaiev, O., Mozhaiev, M., Kuchuk, H.: Method for calculating of R-learning traffic peakedness. In: 2017 4th International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2017, Proceedings, pp. 359-362, DOI: https://doi.org/10.1109/INFOCOMMST.2017.8246416 (2017).
4. Kuchuk, G., Kovalenko, A., Komari, I.E., Svyrydov, A., Kharchenko, V.: Improving Big Data Centers Energy Efficiency. Traffic Based Model and Method. In: Kharchenko, V., Kondratenko Y., Kacprzyk J. (eds) Green IT Engineering: Social, Business and Industrial Applications, Studies in Systems, Decision and Control, vol 171. Springer, Cham, DOI: https://doi.org/10.1007/978-3-030-00253-4_8 (2019).
5. Kuchuk, G., Nechausov, S., Kharchenko, V.: Two-stage optimization of resource allocation for hybrid cloud data store. In International Conference on Information and Digital Technologies, Zilina, pp. 266-271, DOI: http://dx.doi.org/10.1109/DT.2015.7222982 (2015).
6. Kuchuk, G.A., Akimova, Y.A., Klimenko, L.A.: Method of Optimal Allocation of relational Tables. In: Engineering Simulation, 2000, Vol. 17, No. 5, pp. 681 – 689 (2010).
7. Semenov, S., Sira, O., Kuchuk, N.: Development of graphicanalytical models for the software security testing algorithm. In: Eastern-European Journal of Enterprise Technologies,

Vol 2, No 4 (92), pp. 39-46, DOI: https://doi.org/10.15587/1729-4061.2018.127210 (2018).
8. Semenov, S., Voloshyn, D., Lymarenko, V., Semenova, A., Davydov, V.: Method of UAVs Quasi-Autonomous Positioning in the External Cyber Attacks Conditions. Conference Proceedings of 2019 10th International Conference on Dependable Systems, Services and Technologies, 149-153 (2019).
9. Xu, C., Lin, S. B., Fang, J., Li, R. Z.: Prediction-based termination rule for greedy learning with massive data. In: Statistica Sinica, vol. 26, pp. 841-860.
10. François-Xavier Dupé, Sandrine Anthoine: Generalized Greedy Alternatives. In: Applied and Comp. Harmonic Analysis, Elsevier, DOI: https://doi.org/10.1016/j.acha.2018.10.005 (2018).
11. Franti, P.: Efficiency of random swap clustering. In: Journal of Big Data, vol. 5, is. 13, pp. 1–29, DOI: https://doi.org/10.1186/s40537-018-0122-y(2018).
12. Campello, R. J. G. B., Davoud, Moulavi, Arthur, Zimek, Jörg, Sander.: Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. In: ACM Transactions on Knowledge Discovery from Data, vol. 10, is. 1, pp. 1–51.
13. Samuel D.: Pimentel Choosing a Clustering: An A Posteriori Method for Social Networks. In: Journal of Social Structure, vol. 15, Is. 1, DOI: https://doi.org/10.21307/joss-2019-0222014 (2014).
14. Marco, Dorigo: Ant colony optimization. Cambridge, Mass.: MIT Press (2004).
15. Liu, Y.X.; Gao, C., Zhang, Z.L., Lu, Y., Chen, S., Liang, M., Tao, L: Solving NP-hard problems with Physarum-based ant colony system. In: IEEE/ACM Trans. Comput. Biol. Bioinform, 14, 108–120 (2017)
16. A. Tikhomirov, N. Kinash, S. Gnatyuk, A. Trufanov, O. Berestneva et al, Network Society: Aggregate Topological Models, Communications in Computer and Information Science. Verlag: Springer International Publ, Vol. 487, pp. 415-421, 2014.
17. Ruan, Jian Xue Bao: Generating covering arrays using ant colony optimization, exploration and mining. In: J. Softw. 2016, 27, 855–878 (2017).
18. M. Zaliskyi, R. Odarchenko, S. Gnatyuk, Yu. Petrova. A.Chaplits, Method of traffic monitoring for DDoS attacks detection in e-health systems and networks. CEUR Workshop Proceedings, Vol. 2255, pp. 193-204, 2018.
19. S. Gnatyuk, V. Kinzeryavyy, M. Iavich, D. Prysiazhnyi, Kh. Yubuzova, High-Performance Reliable Block Encryption Algorithms Secured against Linear and Differential Cryptanalytic Attacks, CEUR Workshop Proceedings, Vol. 2104, pp. 657-668, 2018.
20. Iavich M., Gagnidze A., Iashvili G., Gnatyuk S., Vialkova V. Lattice based Merkle, CEUR Workshop Proceedings, Vol. 2470, pp. 13-16, 2019.