Controller Placement in SDN with Low Latency using Meta-heuristic Algorithms

Mohammad Erfan Mehrabian[1*], Reza Gholamrezaei[2]
1- Department of Computer Engineering, Faculty of Sciences, Kerman Branch, Islamic Azad University, Kerman, Iran.
Email: me1996mehrabian@yahoo.com (Corresponding author)
2- Department of Computer Engineering, Faculty of Sciences, Kerman Branch, Islamic Azad University, Kerman, Iran.
Email: gholamrezaei@iauk.ac.ir

Received: April 2021 Revised: June 2021 Accepted: July 2021

ABSTRACT:
Software-Defined networks (SDNs) are a new generation of computer networks that have eliminated many of the problems of traditional networks. These networks use a three-tier architecture in which the physical layers, controller, and management are located at different levels. This new architecture has made the network very dynamic, and many of the previous problems in the network have been solved. As the size of the network increases, using a controller across the network will cause issues such as increasing the average latency between the switches and the controller, as well as forming a bottleneck in the controller. For this reason, it is recommended to use multiple physical controllers on the control plane. Due to the cost of purchasing and maintaining the controller, it is necessary to solve the mentioned problem with the least controllers. The question is, to achieve a goal such as reducing latency to an acceptable threshold, at least how many controllers are needed, where the controllers should be located, and which switches should be monitored by which controller? Since this is an NP-Hard problem, methods based on meta-heuristic algorithms can be effective in solving it. In this article, we have solved the problem of controller placement in software-based networks to reduce latency using the cuckoo meta-heuristic algorithm. The simulation results show that the efficiency of our proposed method is between 16 to 70 percent better than the method proposed by the PSO algorithm.

KEYWORDS: Software-Defined Networks, Controller, Controller Placement, Delay, Cuckoo Algorithm.

1. INTRODUCTION

Software-defined networks have been introduced as an emerging phenomenon in network architecture and today, like the cloud, they are a hot topic in the IT[1] world. Transformation in networks is formed by several important factors: a) providing new and better services, b) advancement of technology, c) increasing bandwidth and reducing costs, which create new architectural requirements for networks [1]. SDN[2] or software-Defined networks try to increase the intelligence of networks and by transferring the data control section from the control of hardware switch and router to virtual network software layers and using a centralized software controller, capabilities Provide networking, scalability, flexibility, automation, intelligence, and network software development by organizations Software-defined networks are a new type of network that separates the control layer from the data transfer layer and the network logically In these networks, the control layer manages the network optimally with a single view of the entire network. The control layer does this by using controllers who are the masterminds of the network. The controller can function as a set of distributed controllers that provide a single view of the network [2]. SDN uses include:

- Reduce costs
- Increase productivity
- Performance optimization
- Infrastructure flexibility
- Dynamic and instant performance
- High stability
- Quick smart tracking

SDN consists of separating the data plane from the control plane. Managing the control plane in an SDN network is the responsibility of the logically centralized controller. Given the importance of controllers in SDN architecture and the diversity of architecture and implementation in the market and research areas, there

[1] Information technology
[2] Software Defined networks

Paper type: Research paper
DOI: https://doi.org/10.52547/mjtd.10.3.97
How to cite this paper: M. E. Mehrabian, R. Gholamrezaei, "**Controller Placement in SDN with Low Latency using Meta-heuristic Algorithms**", Majlesi Journal of Telecommunication Devices, Vol. 10, No. 3, pp. 97-104, 2021.

is a need to evaluate and benchmark all of these choices with different performance indicators. A variety of controllers have been implemented with processes of several thousand currents per second to several million currents per second, with multiple language methods, architectures, API[3]s and protocols [3]. Software-Defined networks are a new architecture in computer networks in which network intelligence is logically concentrated in the software controller, and network hardware becomes a simple means of navigation that is closed from they can be programmed through an open interface and have a significant advantage over traditional approaches due to having the following features:

Centralized vision: Includes general network information such as network resource constraints and dynamic changes in network status and information. (QoS[4] general application) [4]

such as programming requirements without the need to control infrastructure elements separately, e.g. Open Flow switches on the data plane, can be actively programmed and reprogrammed dynamically by the central controller to allocate network resources optimally to avoid congestion and improved QoS performance [5].

Openness: that data plane elements (such as Open Flow switches), regardless of the manufacturer, have a single interface that the controller can program the data plane and collect network status.

Open Flow systems can make flow management more flexible and efficient [6].

Due to the separation of the control part from the data part, we need a communication protocol between them, which is commonly used by the Open Flow protocol. The Open Flow protocol is a southern interface protocol, and the devices that support this protocol are commonly called Open Flow-Enabled. The controller is the beating heart of SDNs because it is the core of NOS[5]. The controller is placed between the network equipment (bottom layer) and applications (top layer). An SDN controller is responsible for handling each current is in the network, and it does this by creating current interferences in each switch [7]. Since the controller is the mastermind of the network, and if it is not efficient, it can have very undesirable consequences for the network. It can be said that having an efficient controller in software-based networks is a fundamental element. Therefore, selecting the appropriate number of controllers and their location in the network is one of the challenges of software-based networks that have a significant impact on network productivity. Also, another problem of software-based networks is the latency of these networks. Given that each idea will have its challenges and issues, software-based networks are no exception to this rule; Challenges are inevitable [8]. Most of the efforts made in this field are around reducing the latency between controllers and also reducing the latency between controllers and switches. Therefore, in this study, we try to determine the optimal location of controllers in the network using the cuckoo meta-heuristic algorithm and examine the network based on criteria by examining popular features, controllers in terms of throughput, memory consumption, scalability and response time. Evaluate the following:

Transit: The controller's ability to deliver traffic in response to packets received by the switch.

Response time: The time it takes for the controller to send a response to an input packet.

Scalability: Today, due to the rapid growth of networks, increasing the number of switches and increasing traffic load, the need for scalability of networks in both data part and control part is strongly felt.

Resource efficiency: Software-defined networks by separating the control and data parts, thus easier management and planning to improve the use of resources in the network, including CPU, RAM, memory, etc. [9],[12].

By meeting each of these criteria, it can be claimed that the needs of most networks are met and in fact cover the needs of network policies.

This article consists of 5 sections: section 1 briefly explains the motivation of the research and research problem. Section 2 presents an overview of the relevant controller placement and its issues. Section 3 explains our proposed locating method in detail. Section 4 presents our simulation results and Discussion. Finally, we conclude our work in section 5.

2. LITERATURE REVIEW

In [13], they discuss several aspects of the controller placement problem in terms of resistance and failure tolerance. The delay between controllers is only a small part of multi-objective optimization and has not been studied in depth. In [14], they examine the issue of placing the controller in the SDN to maximize the SDN reliability of the control networks. This research provides a metric to describe the reliability of control networks and also determines the effect of the number of controllers on the reliability of control networks. To solve the problem, he proposes two heuristic algorithms, which are the greedy algorithm and the annealing simulation algorithm, respectively. In [15], they also examined the delay for the controller placement and used clustering; the clustering used by them is an optimized algorithm of k-means. In [16], they examined the delay parameter for the controller placement and

[3] Application Programming Interface

[4] Quality Of Service

[5] Network Operating System

used the density-based clustering method for this study. Unlike other methods, in this method, the structure of the network topology is studied, and then the network is divided into several subnets. The switches within each subnet will be highly interconnected and their connection to other subnet switches will be minimal. In [17], they present a distributed decision approach to evaluate the resilience of software-driven networks (SDNs) to system controller failures. In this regard, a combined method based on switch migration mechanisms (SM) and backup controller (Backup) is used. Defective supplier to the most suitable controller. In [18], they have proposed a way to balance reliability and latency in software-based networks using the proper placement of controllers. This research focuses on the latency and reliability criteria in SDN networks. The meaning of latency in this article is the delay in responding to the data path request, which has a significant effect on the network's latency. This paper shows that the number of controllers and their SDN networks location can affect the two criteria of reliability and latency in SDN networks.

3. MATERIAL AND METHODS

Placing controllers in software-defined networks to reduce latency involves several different phases, the steps of which are shown in the flowchart in Fig. 1. For this purpose, the desired topology must first be extracted from the set of standard topologies available in the Internet Topology Zoo database [19]. After extracting the topology, the desired information must be extracted from it again to form a network graph. This information includes the geographic information of each node, information about network links and the like. Then you can create a graph that represents the network topology. This graph includes vertices, edges, and, if necessary, the location of vertices on a two-dimensional plane. Once the graph is formed, it may be essential to pre-process it. For example, removing duplicate edges as well as leaving out nodes that are in the form of a single node and are not connected to the network graph. After performing these steps, with the help of the cuckoo algorithm, the best place to place the controllers in the network can be found.

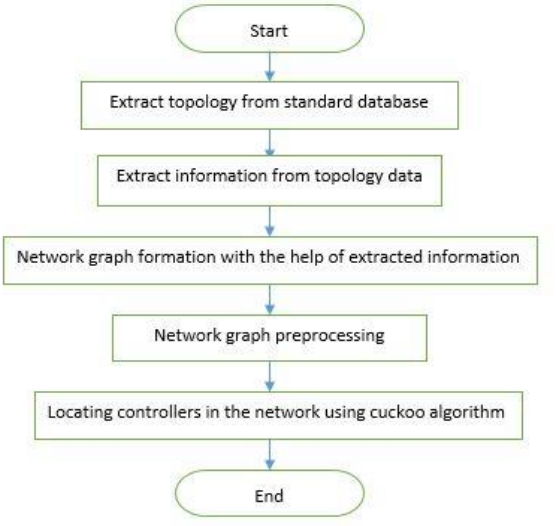


Fig. 1. Basic phases in controller placement in SDN

3.1. Location of Controllers

Before discussing how to use the cuckoo algorithm to locate controllers in SDNs, it is necessary to first provide a brief explanation of the purpose of the work and then to state the problem in the form of relationships that can be solved using meta-heuristic methods.

Consider Fig. 2, this figure represents the network graph of the Bellcanada topology. The extraction and graph formation phases are performed on it. This graph consists of 48 nodes and 64 edges. Each node in this graph represents a switch. The goal is to place the least number of controllers in the network so that the distance between each switch and its controller does not exceed a threshold. The controllers' location can be at the location of each of the network switches, in other words, each controller can be located in one of the 48 positions of the switches. Now, for example, we may want the distance between each switch and its controller not to exceed 3 hops. In this case, the question must be answered: what is the minimum number of controllers required to achieve this goal?

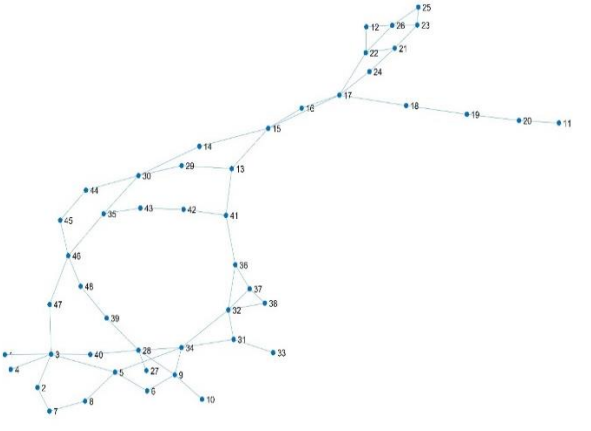


Fig. 2. Bellcanada network graph

3.2. Formulate the Problem

The symbols and defaults used in this article are as follows:

• The communication network is represented by a directionless graph G (V, E).

Sets V and E represent the set of communication nodes and links in the network, respectively.

• Each node $i \in V$ represents a switch. Each switch can accommodate only one controller. Each controller can be located in any of the network nodes. V_c is the set of places where the controllers are located.

y_i = {0,1} is a binary variable in which a value of 1 means that node $i \in V$ hosts a controller and otherwise no.

X_{ij} = {1,0} is a binary variable whose value 1 indicates that the switch $j \in V$ is controlled by a controller in node *i*. Otherwise, it will have a value of 0. Obviously, if $y_i = 1$ then $x_{i,i} = 1$.

• It is assumed that each switch is controlled by a single controller. Suppose $Del_{i,j}$ represents the amount of delay calculated between two nodes $i, j \in V$ so that there is a controller in the node j.

• The purpose of controller placement is to ensure with the least number of controllers that the delay between the switch-controller does not exceed the $\mathfrak{D}$ threshold.

$$\min \quad \sum_{i \in V_C} y_i \qquad (1)$$

s.t.

$$x_{ij} \le y_i \; ; \qquad i \in V, j \in V_C \qquad (2)$$

$$\sum_{i \in V} x_{ij} y_i = 1 \; ; \qquad j \in V_C \qquad (3)$$

$$Del_{i,j} \le \mathfrak{D} \; ; \qquad i \in V, j \in V_C, \; x_{ij} = 1 \qquad (4)$$

Equation (1) indicates that the number of controllers placed in the network should be minimized.

Equation (2) indicates that the switches can only be assigned to a node where the controller is located.

Equation (3) states that each switch in each node, such as j, must be connected exactly to a controller.

Equation (4) shows the latency constraints.

4. RESULTS AND DISCUSSION

In this article, we used the cuckoo algorithm to locate the controllers. This algorithm is one of the high-speed and convenient meta-heuristic algorithms and has a very high exploration power.

4.1. Comparison Algorithm

To prove the superiority of the proposed method, we have used the particle swarm algorithm [20]. The particle swarm algorithm is a well-known and valid algorithm for solving NP-Hard problems. We implemented both algorithms in MATLAB software and considered the same conditions to compare the two algorithms fairly.

4.2. Parameters Studied

Since we use standard datasets to build the network topology, the only input parameter is the allowable distance between the switch and the controller. In this article, we define the distance based on the number of hops between two nodes in a graph without damaging the whole problem. We represent this parameter with the value $MaxHop$. Based on the introduced relations, the value $MaxHop = \mathfrak{D}$ is considered. The higher $MaxHop$ value means that the switch can connect to a longer distance controller. As a result, the threshold ($MaxHop$) increases, fewer controllers are required and conversely as this value decreases, the switch is forced to connect to a nearby controller and the number of controllers required increases.

For a global comparison, we selected 20 standard topologies in the Internet Topology Zoo database in different sizes from small to very large. We also set the value of the $MaxHop$ variable between 1 to 7. Thus, the amount of delay in all 20 topologies from 1 hop to 7 hops was measured in both methods. As a result, 140 implementations of the cuckoo algorithm and 140 implementations of the particle swarm algorithm were generated to have a complete view of the behavior of these two algorithms.

4.3. Overview

Chart. 1 shows the total number of sensors in 20 topologies based on the latency threshold of the total number of controllers used in both methods on 20 different topologies with the number of hops 1 to 7. As can be seen in this figure, if the delay threshold is equal to one hop to the controller, the total of 20 topologies in the cuckoo method requires a total of 382 controllers and in the PSO[6] method requires 454 controllers. This means that the proposed method saved about 16 percent in the number of controllers. In other cases, the delay threshold has been increased the success of the proposed method has been increased so that when the delay value reaches to the threshold 7 hops and the controllers can cover a wider range of sensors, the number of controllers required in the proposed method is about 70 percent less

[6] Particle Swarm Optimization

than in the PSO method. This indicates that as the situation becomes more complex, the proposed method is more efficient than the PSO method.

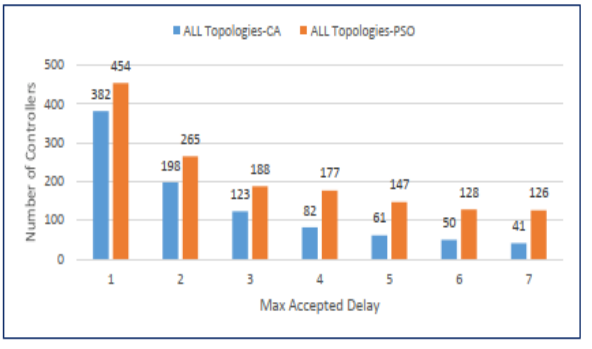


Chart. 1. Total number of sensors in 20 topologies based on delay threshold.

4.4. Comparison of Selected Topologies

In this section, we compare the results of both the proposed method and the PSO method on some topologies with different sizes from small to large.

4.4.1. Comparison of Two Methods in Small Size Topologies

To compare the two methods in a small topology, we chose the Nextgen topology. This topology consists of 17 nodes (switches) and 19 edges (links). Fig. 3 shows the structure of this topology. Chart. 2 shows the number of required controllers in both methods based on the delay threshold, as shown in this figure. It can be seen that with increasing the delay threshold, a smaller number of controllers is needed. In this figure, you can see the superiority of the proposed method over the PSO method in terms of the number of required controllers .For a better understanding of the position of controllers and switches under their command you can see Fig. 4. This figure is based on the delay threshold of $MaxHop$ = 3.

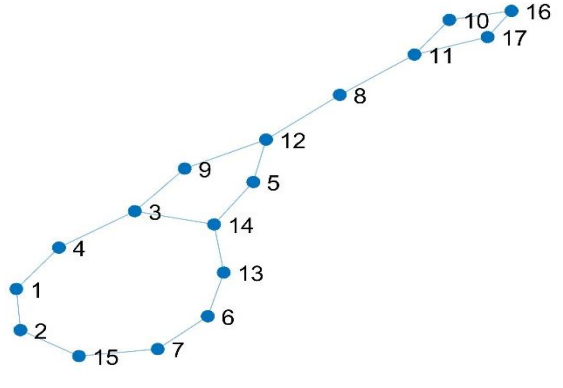


Fig. 3. Nextgen topology.

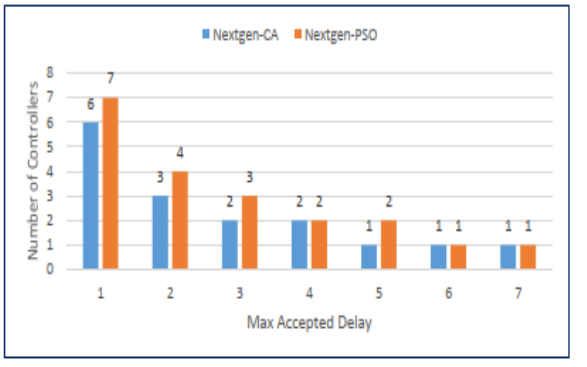
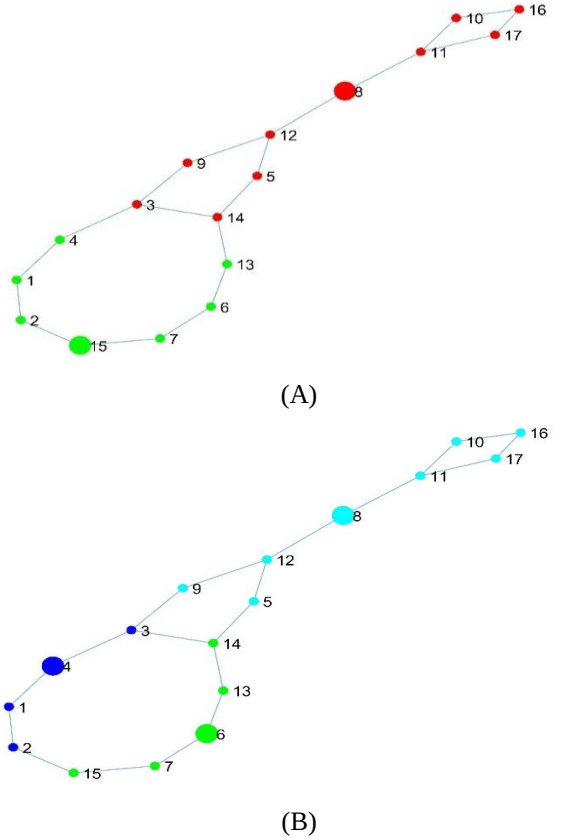


Chart. 2. Number of required controllers based on delay threshold in the proposed method and PSO

.

In fig. 4, you can see how the controllers and the switches under their command are positioned in both methods. The large circles in this figure indicate the location of the controllers and small circles show the location of the switches. Each controller operates its own switches of the same color. As shown in Section A of this figure, the proposed method can locate the controllers with the help of two controllers. In contrast, Section B shows the output of the PSO method in controller placement and requires three controllers.



(A)

(B)

Fig. 4. Number of required controllers based on delay threshold in the proposed method and PSO.

4.4.2. Comparison of Two Methods in Medium Size Topology

To compare the two methods in a medium-sized topology, we chose the Missouri topology. This topology consists of 67 nodes (switches) and 83 edges (links). Fig. 5 shows the structure of this topology. This topology is considered a medium topology in term of size. Controller placement in this group of topologies is more difficult than the first.

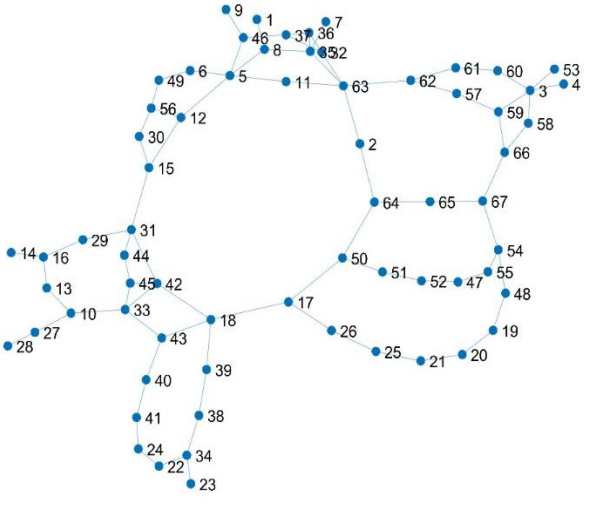


Fig. 5. Missouri topology.

Char. 3, shows the number of required controllers in both methods based on the delay threshold $MaxHop$ = 3.

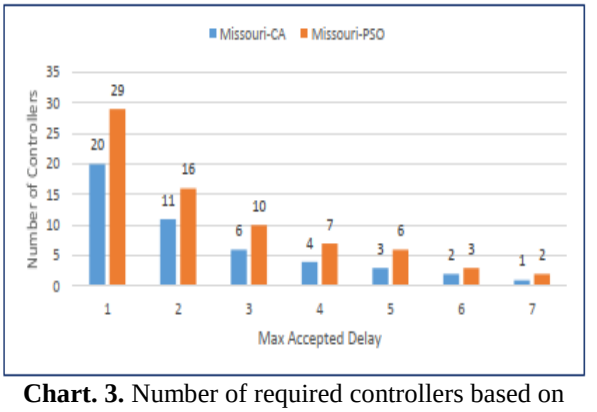
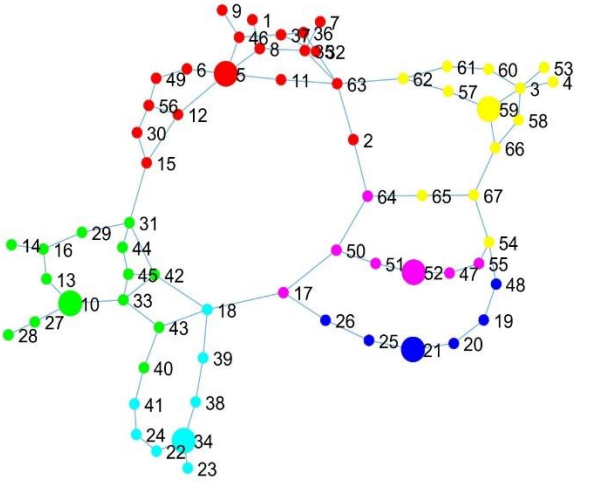


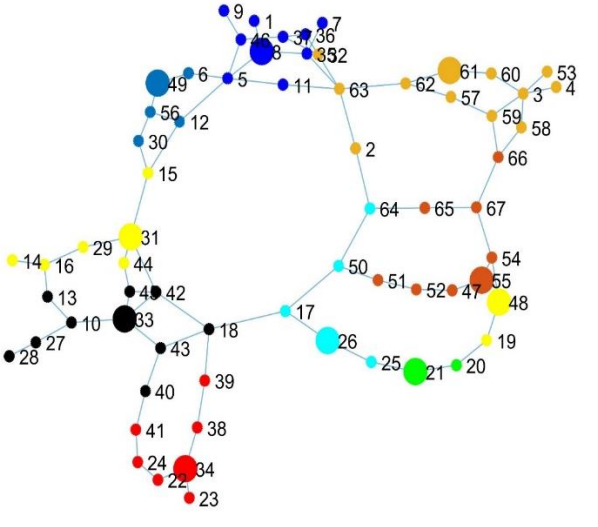
Chart. 3. Number of required controllers based on delay threshold in the proposed method and PSO.

In fig. 6, you can see how the controllers and the switches under their command are positioned in both methods. The value $MaxHop$ = 3 is selected. And to show the position of the controllers and switches under their command, the nodes are colored according to the previous figures. Section A can be seen from this figure, the proposed method can locate the controllers with the help of 6 controllers, while part B of this figure shows the output of the PSO method in controller placement and requires 10 controllers.

As can be seen in this figure, in this topology, too, a smaller number of controllers is required as the delay threshold increases. In this figure, you can see the superiority of the proposed method over the PSO method in terms of the number of required controllers.



(A)



(B)

Fig. 6. Number of controllers required based on delay threshold in the proposed method and PSO.

4.4.3. Comparison of Two Methods in Large Size Topologies

To compare the two methods in a large-sized topology, we chose the Cogentco topology. This topology consists of 197 nodes (switches) and 243 edges (links). Due to the size of this graph, it was not possible to display it. Chart. 4 shows the number of required

controllers in both methods based on the delay threshold. As can be seen, in this topology, as the delay threshold increases, the number of required controllers will decrease, as can be seen, as the delay threshold increases, the results of the proposed method gradually improve, indicating that the proposed method is superior to the PSO method.

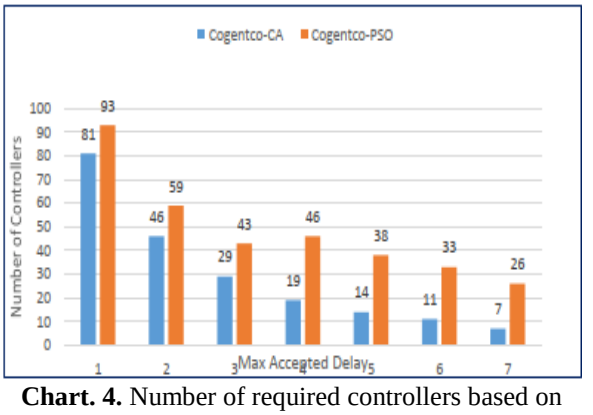


Chart. 4. Number of required controllers based on delay threshold in the proposed method and PSO.

5. CONCLUSION

The location of controllers in software-Defined networks is a very important and fundamental issue in these networks. Numerous and sometimes contradictory goals can be considered for picking controllers in software-defined networks. Goals such as load balancing, increased reliability, repairability, and latency are examples of these goals, the last of which, delay, is of higher importance than others. However, controller placement for any purpose is an NP-Hard issue and can be solved by methods such as estimation algorithms or meta-heuristic methods. In this article, we used one of the newest meta-heuristic algorithms to solve the problem of controller placement in SDN networks, which also has a very high efficiency. The cuckoo algorithm, while simple, has a very high ability to solve NP-Hard problems. For the first time, based on the objective function we designed, we used this meta-heuristic algorithm to locate controllers in software-based networks. To evaluate the efficiency of the proposed method, a number of standard topologies in small to large sizes were used and the results were compared with the results of the PSO algorithm. The simulation results confirmed the superiority of the cuckoo method over the known PSO algorithm.

REFERENCES

[1] Javadpour, A. (2020). **"Providing a way to create balance between reliability and delays in SDN networks by using the appropriate placement of controllers"**. Wireless Personal Communications, 110(2), 1057-1071.

[2] Ivanov, I. G., Hristov, G. V., & Stoykova, V. D. (2021). **"Algorithms for optimizing packet propagation latency in software-defined networks"**. In IOP Conference Series: Materials Science and Engineering (Vol. 1031, No. 1, p. 012072). IOP Publishing.

[3] Das, T., & Gurusamy, M. (2018, July). **"INCEPT: INcremental ControllEr PlacemenT in software defined networks"**. In 2018 27th International Conference on Computer Communication and Networks (ICCCN) (pp. 1-6). IEEE.

[4] Champagne, S., Makanju, T., Yao, C., Zincir-Heywood, N., & Heywood, M. (2018, July). **"A genetic algorithm for dynamic controller placement in software defined networking"**. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 1632-1639).

[5] Zhang, B., Wang, X., & Huang, M. (2018). **"Multi-objective optimization controller placement problem in internet-oriented software defined network"**. Computer Communications, 123, 24-35

[6] Abdi Seyedkolaei, A., Hosseini Seno, S. A., & Moradi, A. (2021). **"Dynamic controller placement in software-defined networks for reducing costs and improving survivability"**. Transactions on Emerging Telecommunications Technologies, 32(1), e4152.

[7] Jalili, A., Keshtgari, M., & Akbari, R. (2020). **"A new framework for reliable control placement in software defined networks based on multi-criteria clustering approach"**. Soft Computing, 24(4), 2897-2916.

[8] Mohanty, S., Priyadarshini, P., Sahoo, S., Sahoo, B., & Sethi, S. (2019, October). **"Metaheuristic Techniques for Controller Placement in Software-Defined Networks"**. In TENCON 2019-2019 IEEE Region 10 Conference (TENCON) (pp. 897-902). IEEE.

[9] Mbodila, M., Isong, B., & Gasela, N. (2020, November). **"A Review of SDN-Based Controller Placement Problem"**. In 2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC) (pp. 1-7). IEEE.

[10] Jalili, A., Keshtgari, M., Akbari, R., & Javidan, R. (2019). **"Multi criteria analysis of controller placement problem in software defined networks"**. Computer Communications, 133, 115-128.

[11] Rasol, K. A., & Domingo-Pascual, J. (2020, September). **"Multi-level Hierarchical Controller Placement in Software Defined Networking"**. In International Networking Conference (pp. 131-145). Springer, Cham.

[12] Syed-Yusof, S. K., Numan, P. E., Yusof, K. M., Din, J. B., Marsono, M. N. B., & Onumanyi, A. J. (2020, December). **"Software-Defined Networking (SDN) and 5G Network: The Role of Controller Placement for Scalable Control Plane"**. In 2020 IEEE International RF and Microwave Conference (RFM) (pp. 1-6). IEEE.

[13] Hock, D., Gebert, S., Hartmann, M., Zinner, T., & Tran-Gia, P. (2014, May). **"POCO-framework for Pareto-optimal resilient controller placement in SDN-based core networks"**. In 2014 IEEE Network Operations and Management Symposium (NOMS) (pp. 1-2). IEEE.

[14] Hu, Y., Wang, W., Gong, X., Que, X., & Cheng, S. (2014). **"On reliability-optimized controller placement for software-defined networks"**. China

Communications, 11(2), 38-54.

[15] Wang, G., Zhao, Y., Huang, J., Duan, Q., & Li, J. (2016, May). **"A K-means-based network partition algorithm for controller placement in software defined network"**. In 2016 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

[16] Liao, J., Sun, H., Wang, J., Qi, Q., Li, K., & Li, T. (2017). **"Density cluster based approach for controller placement problem in large-scale software defined networkings"**. Computer Networks, 112, 24-35.

[17] Vosoughi, Mahsa; Shahram Jamali and Masoud Bakravi, (1398), **"Presenting a Distributed Decision Approach to Assess the Tolerance of Software-Based Networks (SDNs) to System Controllers Failure"**. 8th National Conference on Computer Science and Engineering, and Information Technology, Babol, Scientific Research Institute of Alamavaran Danesh.

[18] 18 Navaei, Hamidreza and Mohammadreza Majmeh, (1396), **"Presenting a method to balance between reliability and latency in software-based networks using the proper placement of controllers"**. Fourth National Conference on Information Technology, Computer and Telecommunications, Mashhad, Torbat Heydariyeh University .

[19] Maity, I., Dhiman, R., & Misra, S. (2021). **"MobiPlace: Mobility-Aware Controller Placement in Software-Defined Vehicular Networks"**. IEEE Transactions on Vehicular Technology, 70(1), 957-966.

[20] Rawat, D. B. (2019). **"Fusion of software defined networking, edge computing, and blockchain technology for wireless network virtualization"**. IEEE Communications Magazine, 57(10), 50-55.