Computer Assisted Methods in Engineering and Science
30(4): 539–556, 2023, doi: 10.24423/cames.498

This article belongs to the *Special Issue on Computer Aided Software Design for Multi-Concern Assurance* edited by Dr. J.I.-Z. Chen, Dr. K.-L. Du and Dr. H. Wang

A Hybrid Evolutionary Algorithm of Optimized Controller Placement in SDN Environment

J. HEMAGOWRI*, P. TAMIL SELVAN

Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore, India; e-mail: tmselvanin@gmail.com

* *Corresponding Author e-mail: hemagowri.j@gmail.com*

Controller placement problem (CPP) is a significant technological challenge in software defined network (SDN). Deployment of a properly designed SDN-based network is required to detect optimal number of controllers for enhancing the network's performance. However, the best possible controller placement for enhancing the network's performance faces many issues. To solve the CPP, a novel technique called the hybrid evolutionary algorithm of optimized controller placement (HEA-OCP) in SDN environment is introduced to increase network's performance by different network topologies. In the proposed model, optimized controller placement using improved multi-objective artificial fish optimization is employed to improve data transmission and reduce latency. Controller placement can be determined using an undirected graph based on a variety of factors, including propagation delay, load balancing capabilities and bandwidth, fault tolerance and data transfer rate, and a variety of other factors. For each controller, the fitness value is calculated over multi-criteria functions. The optimizer's performance can be improved with the use of Gaussian chaotic maps. In large-scale SDN networks using HEC-OCP, the algorithm dynamically analyzes the optimal number of controllers and the best connections between switches and controllers. As a result, the overall network performance is improved and the delay minimization-based controller placement strategy is obtained. The simulation of HEA-OCP with existing methods is conducted by a network topology dataset of various metrics, namely packet delivery ratio, packet drop rate, throughput, average latency, and jitter. The proposed HEA-OCP improves the packet delivery and throughput with reduced average latency, and packet drop ensures more instantaneous communications in real-time applications of SDN for better decision-making.

Keywords: controller placement problem, software defined network, Gaussian chaotic map, multi-criteria fish swarm optimization.

1. Introduction

SDN decouples the control plane from the data plane. SDN controllers are adequate for this task as they provide a single, unified surface. However, a single controller controlling all the switches in a large network will increase the switch-to-controller latency. Therefore, an optimum number of controllers are necessary in SDN. SDNs have several applications such as security services, network intelligence and monitoring, compliance and regulation-bound applications, distributed application control and cloud integration, and many high-performance applications such as geographic information system (GIS), engineering, computer-aided design (CAD), etc. [1]. Multiple physical controllers provide a logically centralized control plane in SDN, which is one of its distinguishing features. There is a critical design challenge known as the controller placement problem (CPP) in the SDN environment. It has an effect on the network's latency, flow setup time, network availability, controller load balancing, and energy consumption.

Almost every element of SDN, from state distribution choices to fault tolerance to network performance, is affected by the CPP, which has launched a lot of studies [2]. The decoupled control plane's functionalities and the network's state distribution choices are both affected by the CPP. The CPP presents itself as trivial issue that is primarily about selecting the number and location of controllers to be placed in the network. This involves: (i) choosing the optimal number of controllers for a particular network, (ii) the location of these controllers in a certain area, and (iii) assigning function of controllers to switches and designing an efficient and accurate control plane inside the network. In the following paragraphs, brief summaries of the most significant research approaches of CPP in SDN are given.

A simulated annealing-based failure foresight capacitated CPP (SA-FFCCPP) was introduced in [3] to minimize latencies from all switches to the respective backup controllers. The placement of controllers in this model was not considered for the load balancing capability to minimize packet loss. A garter snake optimization capacitated CPP (GSOCCPP) was developed in [4] achieving lower execution time. But the designed optimization technique failed to consider various factors, such as fault tolerance and bandwidth, in obtaining minimum delay.

A comprehensive mathematical formalization of CPP was presented in [5] for decreasing latency. However, the performance of the latency-aware CPP was not considered. A parameter optimization model (POM) was introduced in [6] for CPPS with minimum delay. But it failed to obtain reasonable SDN controller placement method.

Varna-based optimization (VBO) was introduced in [7] for consistent controller issue that minimizes total average latency. However, the developed method was unsuccessful in considering numerous constraints to solve CPP in SDN. Fault-tolerance meta-heuristic-based scheme was developed in [8] for CPP, maximizing the network connectivity and load balancing among the controllers. But, the performance of latency was not minimized.

A scalable algorithm was designed in [9] for controller placement issues on large-scale networks. The developed algorithm decreased the latency, but a higher throughput was not achieved. Two new maximum entropy-based clustering techniques were introduced in [10] for controller placement. However, the heuristic technique was unsuccessful in improving optimal controller placement. Efficient meta-heuristic-based RALO was developed in [11] for multi-objective controller placements. However, the method was unsuccessful in reducing switch-to-controller delay.

A nature-inspired population-based meta-heuristic algorithm was developed in [12]. But it failed to consider the different network topologies for conducting the performance test. The heuristic algorithm based on Dijkstra and K-means algorithm was developed in [13]. However, the efficient heuristic algorithm was not implemented to enhance the performance of controller placement.

A new controller placement algorithm was designed in [14] for reducing delay and convergence rate. However, the designed algorithm failed to apply other metrics for further minimizing the propagation latency. A CPP was solved in [15] for multi-link failures to decrease the number of controller delays (variables). However, it failed to use the heuristic algorithm to reduce the overheads. A salp swarm optimization algorithm (SSOA) was developed in [16] to improve the optimizer's performance and minimize the execution time. But the network's latency was not minimized.

A reliable CPP model (RCPPM) was introduced in [17] to maximize the reliability of SDN. However, the different objective functions were not considered for enhancing the SDN network. The ant colony system with external memory (ACS-EM) method was introduced in [18] for SDN-based device-to-device communications. Nevertheless, the reliability of the algorithm was not improved.

Near-optimal algorithms were introduced in [19] to minimize the low computational complexity for solving the placement problem. However, the latency-aware CPP was not considered. RCPPM was also presented in [20] to improve the reliability of SDN. However, the designed model failed to consider the multi-objective optimization.

An optimized submodularity-based approach was developed in [21] to solve the CPP with minimum latency. But, the higher data delivery ratio between the switches and controller was not achieved. A new simulated annealing genetic algorithm was designed in for resolving CPP and minimizing latency. Yet, the designed method was not efficient in considering the multiple objective functions.

In order to overcome the existing issues in SDN, a novel HEA-OCP is introduced in this work with the following contributions:

- To solve CPP in SDN, a novel HEA-OCP is introduced based on the multi-criteria functions.

- The proposed HEA-OCP technique finds the optimum controller on the SDN environment based on the metrics such as propagation latency, load balancing capability, bandwidth, fault tolerance, and data transmission rate.
- To better regulate the reduction of local optima, chaotic maps extract random parameters based on a Gaussian distribution. Using unimodal and multimodal optimization, a logistic chaotic map is determined to be the best fit for the problem at hand.
- Finally, a simulation test is conducted with the network topology for verifying HEA-OCP and existing methods. HEA-OCP is explained with various parameters.

The article consists of the following sections. Section 2 presents a description of the novel HEA-OCP through a different process. Section 3 introduces the simulation settings with the network topology. Section 4 provides the results and discussions for three different methods. Section 5 presents the conclusion of this article.

2. Methodology

SDN is a decoupled structural design that enables an administrator to make a customizable and controllable network. In the control plane, controllers are required to calculate and direct switches. By the SDN scenario, choosing an optimal number of controllers with operation location is termed a controller issue. The HEA-OCP is introduced based on the multi-criteria functions for controlling the controller issue. By replicating the emergent behavior of biological swarms, swarm intelligence (SI) algorithms have been shown to be a complete solution for solving complicated optimization problems. Data science is gaining popularity, and this necessitates the administration and quick analysis of large amounts of data. Because of their dynamic features, device mobility, wireless connection, and information supply, SI's intelligent algorithms can address the complex issues of IoT systems. In this work, the SI algorithm helps to avoid overfitting problems.

Figure 1 describes the basic block diagram of the proposed HEA-OCP technique to solve the CPPs. The switches are represented in the red, the controllers are in blue, and the lines are referred to as links in the network topology. The controller placement is organized into an undirected graphical model $G(s, e)$ where s indicates switches and e indicates a link between switches. A number of controller's $C = \varphi_1, \varphi_2, ..., \varphi_n$ are located in an optimal way in the SDN-based infrastructure. To improve the speed of data transmission, an optimum position of the controller is identified based on multi-objective functions, such as propagation latency (α_{lat}), load balancing capability (α_{load}), bandwidth (α_{bw}), fault tolerance (α_{ft}), and data transmission rate (α_{DTR}).

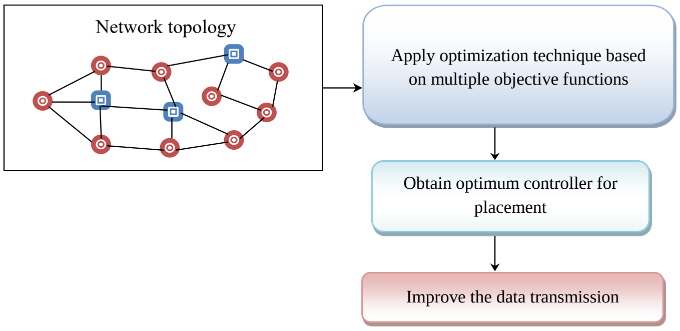
Network topology
Apply optimization technique based on multiple objective functions
Obtain optimum controller for placement
Improve the data transmission

Fig. 1. Basic block diagram of the proposed HEA-OCP technique.

By applying the proposed HEA-OCP technique, the oppositional learned multi-objective artificial fish optimization is a swarm intelligence processed based on the population and stochastic search. In contrast to the existing optimization algorithm, the proposed algorithm is the best technique. Oppositional learned artificial fish swarm is a meta-heuristic technique based on behaviors, such as prey, swarm, and follow. Instead of relying on one other for knowledge, individuals can allocate more of their limited mental resources to other activities, which can lead to higher levels of cognitive performance when working together. Perceived competition for food, which is more prevalent in larger groups when foraging is involved, can cause this effect. It has been widely accepted in previous studies that fish shoals have better cognitive abilities because of this individual-level explanation. It is possible to use the sigmoid function for various fish movements in a swarm to better detect movement and community detection in a swarm. Here, the artificial fish was associated with the number of controllers, and food supply represents the multi-objective functions, such as propagation latency (α_{lat}), load balancing capability (α_{load}), bandwidth (α_{bw}), fault tolerance (α_{ft}), and data transmission rate (α_{DTR}). Opposition learning is applied to select better individuals. The main aim of the opposition-based learning concept is to consider the estimates or actions that attempt the opposite for enhancing accuracy with lesser time complexity.

The optimization introduces a population of n number of artificial fishes (i.e., controllers) in the search space (i.e., SDN),

$$C = \varphi_1, \varphi_2, \ldots, \varphi_n. \tag{1}$$

After the initialization, the opposition-based learning concept is applied to generate an opposite population for obtaining the global optimum solution compared to an arbitrary one. Hence, the opposite population is

$$C' = a_i + b_i - C, \tag{2}$$

where C indicates the current population, where a_i and b_i symbolize the smallest and highest value of dimensions, respectively. After opposite and current population generation, fitness is measured based on multi-objective functions.

In networking, the propagation latency is calculated by the time taken to transmit data via the controller to the switches. It is measured as:

$$\alpha_{\text{lat}} = \text{time}\,(\text{TD}), \tag{3}$$

where α_{lat} denotes a latency, TD is the data transfer from the controller to the switches. Load balancing capability (α_{load}) between the switch and controllers is measured based on the load factor. It is referred to as the proportion of load to the highest requirement at a particular time,

$$L_f = \frac{\operatorname{avg} L}{\max D}, \tag{4}$$

where L_f indicates a load factor, $\operatorname{avg} L$ represents an average load, and $\max D$ indicates a maximum demand. A load factor less than 1 is represented by a controller that has better load balancing capability.

Bandwidth is the significant objective function denoted as the maximum rate of data transfer capacity in a particular amount of time from the controller to the switches,

$$\alpha_{\text{bw}} = \frac{\max D}{S}, \tag{5}$$

where α_{bw} indicates a bandwidth, $\max D$ [bits] represents the maximum rate of data transfer in the unit bits, and S denotes time in second. Therefore, the bandwidth is measured in terms of mega or gigabits per second [Mbps or Gbps].

Fault tolerance is the ability of a controller to maintain the operation properly when a failure occurs. The failure rate is calculated by the proportion of number of failures that occurred, to the entire operating time. It is formulated as follows:

$$R_f = \frac{\text{number of failures}}{\text{total operating time}}, \tag{6}$$

where R_f indicates a failure rate. When the minimum failure rate occurs, the controller has better fault tolerance capability. The data transmission rate is the significant metric defined as the amount of data transmitted over a channel in a particular unit of time. The unit of data transmission rate is bits/s.

Based on the above-said resources, the optimal location of controller placement is identified through the fitness measure. The fitness is measured as:

$$\beta_F = [\arg\min \alpha_{\text{lat}}]\ \&\&\ [\arg\max \{\alpha_{\text{load}}, \alpha_{\text{bw}}, \alpha_{\text{ft}}, \alpha_{\text{DTR}}\}], \tag{7}$$

where β_F indicates a fitness function, arg min denotes an argument of a minimum function, and arg max indicates an argument of a maximum function. The controller with minimum latency and maximum load balancing capacity, bandwidth, fault tolerance and data transfer rate are selected as optimal. Where '&&' indicates the logical 'AND' operation and this operation must satisfy both conditions.

Next, current and opposite populations are collective, and the categorization of fishes is performed by fitness value. Lastly, n greatest artificial fishes were chosen via population for additional transforming. Based on the fitness value, different behaviors of the artificial fish's positions, such as search or prey, swarm, and follow, are performed to discover the best global solution.

2.1. Search or prey behavior

In the proposed HEA-OCP technique, the search for prey is a fundamental behavior of fish search for food source. The artificial fish find the awareness of food in water through vision or sense. Let us consider the present point of fish is q_i and the new point of fish is $q_{i(t+1)}$. If the fitness is greater than the other, i.e., $\beta_F(c_i) < \beta_F(c_j)$, the search behavior is executed and updates the new position based on the Gaussian chaotic map. The chaotic act as a method helps to avoid the proposed optimization staying in the local optimum position for a longer time. Therefore, a novel Gaussian chaotic map introduces into a search behavior an algorithm to improve global convergence and search efficiency

$$q_{i(t+1)} = q_i + \vartheta * R * \exp\left(-\frac{1}{2} * \left(\frac{(q_j - q_i)}{\|q_j - q_i\|}\right)^2\right), \tag{8}$$

where $q_{i(t+1)}$ indicates an updated position, q_i represents the current position, R denotes a random number from $(0 < r < 1)$, ϑ specifies a footstep of the fish movement in a random positive number, and $\|q_j - q_i\|$ indicates the visual distance between the position of the j-th fish and the position of the i-th fish.

2.2. Swarm behavior

The second one is the swarm behavior, where all the fishes are combined during the movement to avoid the risks while searching for the food source.

The swarm behavior of the artificial fish is accomplished when the $\beta_F(c_c) < \beta_F(c_i)\&\&\left(\frac{n_n}{n} < \omega\right)$, where $\beta_F(c_c)$ is the fitness of artificial fish at the center position, n_n indicates a current neighborhood, n represents a total number of fish, and ω denotes a crowd factor value from 0 to 1. It means the center fish has a higher fitness value than the other. Subsequently, the position of the artificial fish gets updated based on the center fish,

$$q_{i(t+1)} = q_i + \vartheta * R * \exp\left(-\frac{1}{2} * \left(\frac{\|q_c - q_i\|}{\sigma}\right)^2\right), \tag{9}$$

where $q_{i(t+1)}$ indicates an updated point of fish, q_i represents the current point, R denotes a random number varied from zero to one $(0 < R < 1)$, δ denotes a step of the fish moving, which is a random positive number, ϑ specifies a footstep of the fish movement in a random positive number, and $\|q_c - q_i\|$ is the visual distance between the position of the j-th fish and the central position of the fish in its current neighborhood.

2.3. Follow behavior of the fish

Finally, in the moving behaviors, when a single fish or a number of fish find their food, the neighborhood follows and reaches the food. If $\beta_F(c_c) < \beta_F(c_i)\&\&\left(\frac{n_n}{n} < \omega\right)$, then the following behavior is implemented and the position is updated as follows:

$$q_{i(t+1)} = q_i + \vartheta * R * \exp\left(-\frac{1}{2} * \left(\frac{\|q_{\max} - q_i\|}{\sigma}\right)^2\right), \tag{10}$$

where $q_{i(t+1)}$ denotes the updated point of fish, q_i represents the current point, $q_{\max}$ denotes a position having the best fitness function value, R denotes a random number varied from zero to one $(0 < R < 1)$, ϑ denotes a step of the fish movement which is a random positive number, and $\|q_{\max} - q_i\|$ is the visual distance among point of i fish and point of fish having the best fitness function. Now, replace the older fish with a new optimal one based on fitness. This process is repeated until the maximum iteration gets reached.

Figure 2 illustrates the flowchart of the HEA-OCP to find the best optimal controller for improving the data transmission. The algorithmic process of HEA--OCP is given in Algorithm 1.

The algorithmic process of HEA-OCP in an SDN network. The current population of controllers and the opposite population were initialized arbitrarily. The multi-criteria function is assessed for each controller. The fitness is calculated based on the multi-criteria. The fitness values of the two controller populations were then used to sort them. Afterward, the most recent greatest n controllers are chosen to determine the best possible global solution. After then, the swarm's coordinates are recalculated using a Gaussian chaotic map to identify the world's most effective controllers for dealing with the placemat problem. Once the maximum iteration is achieved, the operation is repeated. Finally, the optimal controller is obtained for improving the data delivery and minimize latency.

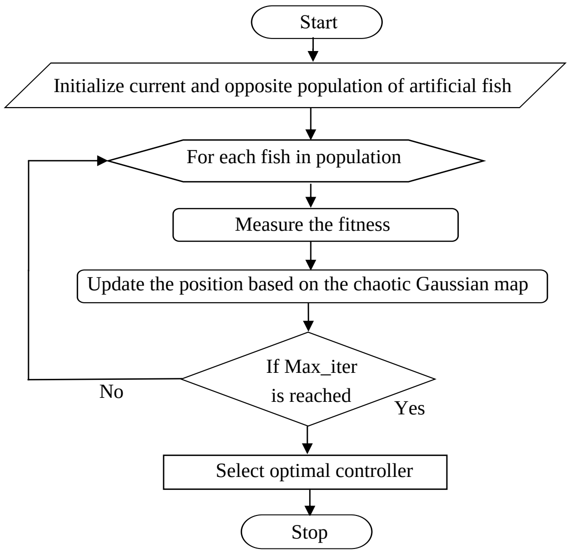
Start
Initialize current and opposite population of artificial fish
For each fish in population
Measure the fitness
Update the position based on the chaotic Gaussian map
If Max_iter is reached
No
Yes
Select optimal controller
Stop

FIG. 2. Flow chart of HEA-OCP.

Algorithm 1. HEA-OCP

Input: Dataset, Number of controllers $C = \varphi_1, \varphi_2, ..., \varphi_n$,
Output: Find optimal controllers for placement

```
    Begin
1.  For each controller $\varphi_i$
2.      Initialize the current population of $C = \varphi_1, \varphi_2, ..., \varphi_n$
3.      Initialize opposite populations $C'$
4.          Calculate the fitness $\beta_F$
5.          Merge the two populations $C$ and $C'$
6.          Sort the controllers based on fitness
7.          Select current best 'n' controllers
8.          While ($t <$ Max_iter)
9.              if ($\beta_F(c_i) < \beta_F(c_j)$) then
10.                 Update the position $q_{i(t+1)}$ using Eq. (10)
11.             else if $\left(\beta_F(c_c) < \beta_F(c_i) \&\& \left(\frac{n_n}{n} < \omega\right)\right)$ then
12.                 Update the position $q_{i(t+1)}$ using Eq. (10)
13.             else if $\left(\beta_F(c_c) > \beta_F(c_i) \&\& \left(\frac{n_n}{n} < \omega\right)\right)$ then
14.                 Update the position $q_{i(t+1)}$ using Eq. (10)
15.             end if
16.         Replace the old controllers with the current best
17.         end while
18.         $t = t + 1$
19. end for
20. Return (best optimal solution)
    End
```

3. Simulation scenario

In this section, a simulation of the HEA-OCP and existing SA-FFCCPP [3] and GSOCCPP [4] is carried out in an NS-2 network simulator using a general Bayesian network (GBN) network topology. This dataset includes simulation results of bandwidth transmitted, the total amount of packets transferred, the total number of packets dropped, and the average delay in every source-destination pair variance of delay (jitter) through packets that are transferred. Simulation parameter settings are presented in Table 1.

Table 1. Simulation parameters.

Simulation parameters	Values
Network simulator	NS2.34
Number of nodes (switches) in GBN network topology	17
Number of data packets	25, 50, 75, 100, 125, 150, 175, 200, 225, 250
Number of controllers	5
Simulation time	300 s
Protocol	DSR
Number of runs	10

3.1. Simulation outcomes

Let us consider the GBN network topology dataset for conducting the simulation with 17 nodes (switches) and 5 controllers.

As shown in Fig. 3, the switches and controller are organized into an undirected graphical model. As shown in Fig. 5, blue color-circulated nodes are called controllers, whereas the red-colored nodes are called switches.

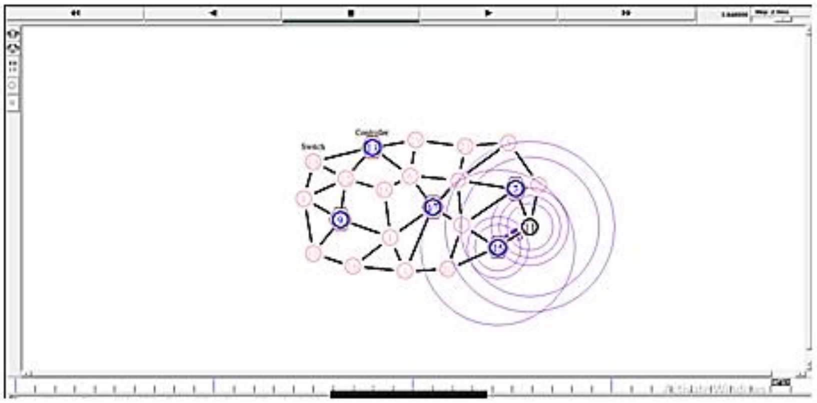

Fig. 3. Switch and controller placement.

Figure 4 shows the optimum controller placements using Gaussian chaotic map multi-criteria fish swarm optimization technique. As shown in Fig. 4 green color circulated nodes 9, 17, and 15 are selected as optimum controller placements.

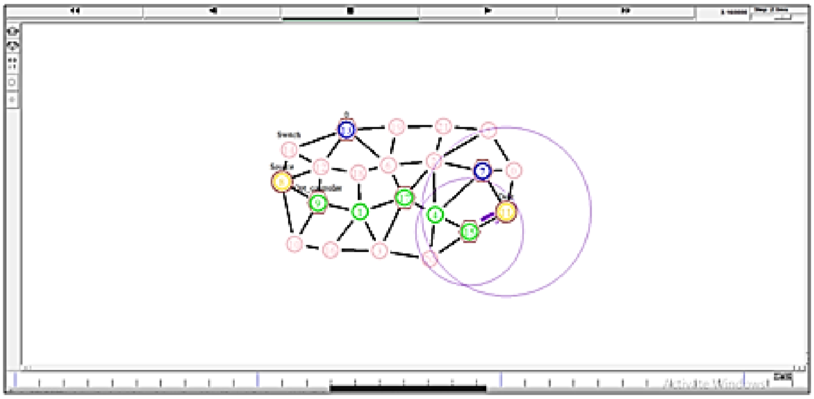

Fig. 4. Optimum controller placement.

Finally, the data transmission is performed via the optimal controller placement from source to destination. As shown in Fig. 5, the yellow-colored node represents the source and destination.

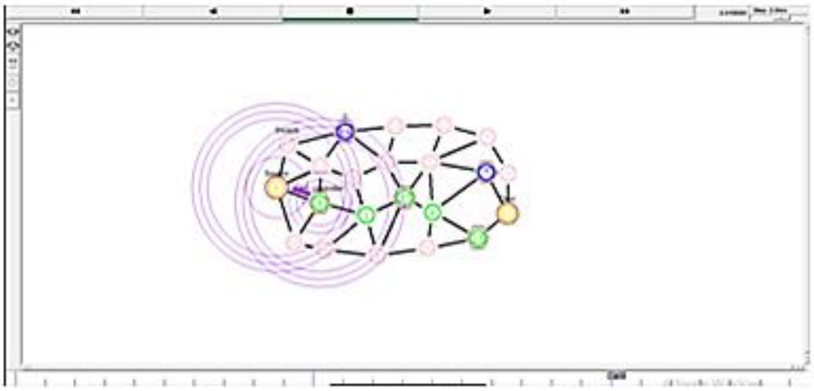

Fig. 5. Data transmission.

4. Comparative results analysis

The experimental results of HEA-OCP, SA-FFCCPP [3], and GSOCCPP [4] are discussed with respect to various performance metrics, namely packet delivery ratio, packet drop rate, throughput, average latency, and jitter.

4.1. Impact of packet delivery ratio

Packet delivery ratio is the number of packets effectively delivered via source to destination pair in the network. It is measured by

$$R_{PD} = \frac{ND}{N} * 100, \tag{11}$$

where R_{PD} indicates packet delivery ratio, N indicates the number of data packets, and ND indicates the number of packets successfully delivered. Packet delivery ratio is calculated in percentage [%].

Table 2 details the packet delivery ratio with different data packets. The amount of data packet counts range of 25, 50, 100, ..., 250. The experimental result of three different methods: HEA-OCP, existing SA-FFCCPP [3], and GSOCCPP [4] are used for calculating the packet delivery ratio. It reveals that the HEA-OCP increases the packet delivery ratio more than the existing methods. The experiment is carried out by 25 data packets, and the packet delivery ratio is 88% by the proposed HEA-OCP, whereas 84% of the packet delivery ratio was obtained in [3], and 80% of the packet delivery ratio was obtained in [4]. For each method, ten outcomes are observed by various numbers of data packets. The average of ten outcomes indicates that the proposed HEA-OCP enhances the packet delivery ratio by 3% and 5%, compared with the other two state-of-the-art methods, respectively.

Table 2. Packet delivery ratio.

Number of data packets	Packet delivery ratio [%]		
	HEA-OCP	SA-FFCCPP	GSOCCPP
25	88	84	80
50	90	88	84
75	92	89	87
100	90	87	85
125	91	90	88
150	89	87	85
175	92	90	89
200	90	88	86
225	92	90	89
250	91	89	86

The packet delivery ratio using three different methods: HEA-OCP, and existing SA-FFCCPP [3] and GSOCCPP [4], is illustrated by the amount of data packets revealed in Table 2, the number of data packets consumed in the horizontal direction and packet delivery ratio observed in the vertical direction. The graphical plot indicates that the packet delivery ratio of the proposed HEA--OCP is higher than the existing methods. Because of this enhancement, it was used to select the optimal controller placement with multi-criteria functions. The optimal controller placement in the network topology enhances data delivery.

4.2. Impact of packet drop rate

The packet drop rate was measured as the number of packets dropped via source to destination pair in the network. The packet drop rate is estimated below:

$$\text{Packet drop rate} = \frac{\text{number of packets dropped}}{N} * 100, \tag{12}$$

where N indicates the number of data packets. It is expressed in percentage [%].

Table 3 presents the performance results of the packet drop rate versus the number of data packets transferred through 25–250. The above results prove that the packet drop rate of HEA-OCP is smaller than the other baseline techniques. For 25 data packets measured in the experiment, the packet drop rate of HEA--OCP is found to be 12%, 16% and 20% using SA-FFCCPP [3] and GSOCCPP [4], respectively. The packet drop rate is reduced by 19% and 32% compared to SA-FFCCPP and GSOCCPP, respectively.

TABLE 3. Packet drop rate.

Number of data packets	Packet drop rate [%]		
	HEA-OCP	SA-FFCCPP	GSOCCPP
25	12	16	20
50	10	12	16
75	8	11	13
100	10	13	15
125	9	10	12
150	11	13	15
175	8	10	11
200	10	12	14
225	8	10	11
250	9	11	14

The packet drop rate of the proposed HEA-OCP is minimized by existing methods. Because of this enhancement, it was used to select the optimum controller placement with higher bandwidth as well as better load balancing capability. The data delivery was improved as well as packet drop was reduced.

4.3. Impact of throughput

It is measured as the number of data packets effectively distributed to the destination in a specific time. Throughput is measured as follows:

$$\text{Throughput} = \frac{\text{amount of data packets delivered}}{\text{time}}. \tag{13}$$

It is computed in bits per second [bps].

Table 4 provides the comparative analysis of the throughput of the three different methods HEA-OCP, SA-FFCCPP [3], and GSOCCPP [4]. By utilizing the proposed HEA-OCP technique, 183 bits of data packets were transferred in 1 second. But the throughput of SA-FFCCPP and GSOCCPP is observed at 171 bps and 160 bps, respectively. This significant improvement is achieved by selecting the optimum controller with higher bandwidth capacity and data transmission rate. Besides, efficient bandwidth-aware data transmission is performed for continuous data flow. The throughput of data transmission was improved from source to destination.

Table 4. Throughput.

Size of data packets [KB]	Throughput [%]		
	HEA-OCP	SA-FFCCPP	GSOCCPP
15	183	171	160
30	259	218	205
45	382	326	310
60	493	412	390
75	595	547	480
90	680	610	560
105	782	720	670
120	859	810	770
135	1050	925	880
150	1263	1035	960

4.4. Impact of average latency

Average latency is referred to as the time consumed for transferring data packets via the controller to switches

$$L_{\text{avg}} = N * t\,(\text{TD}), \tag{14}$$

where L_{avg} indicates an average latency in ms, N indicates the number of data, and $t\,(\text{TD})$ represents time consumed with the algorithm for data transmission.

Table 5 illustrates the average latency with respect to the number of switches. From the observed results, the average latency is considerably reduced by HEA-OCP compared to the other two methods. The overall results indicate that the HEA-OCP technique decreases the average latency by 20% compared to SA-FFCCPP [3] and by 37% compared to GSOCCPP [4].

Table 5. Average latency.

Number of switches	Average latency [ms]		
	HEA-OCP	SA-FFCCPP	GSOCCPP
2	0.18	0.25	0.4
4	0.32	0.5	0.7
6	0.45	0.62	0.8
8	0.67	0.76	0.95
10	0.8	0.98	1.4
12	1.2	1.5	1.7
14	1.6	1.8	2.1
16	2.1	2.3	2.5

From the results, the average latency of the HEA-OCP was comparatively more reduced than the two existing methods. The reason for enhancement is used to apply the Gaussian chaotic map multi-criteria fish swarm optimization to select the controller placement with better load balancing capacity and higher bandwidth. This increases the data delivery with minimum latency.

4.5. Impact of jitter

Jitter defined as the variation of data packet transmission via source to destination in time delay. It is computed by

$$J = |d(i) - d(i-1)|, \tag{15}$$

where J indicates jitter, $d(i)$ denotes a current delay, and $d(i-1)$ indicates a delay of packet transmission. It is measured in milliseconds [ms].

Table 6 indicates that the performance results of the jitter were attained in a variation of delay with the number of data packets transferred from source node 25–250. Jitter is an important parameter employed for detecting the source node that extends the time of data delivered by the destination. HEA-OCP is reduced by baseline methods. Jitter improves all three methods owing to which the packet counts are improved. For the experiment, 25 data packets measured the jitter of 6 ms utilizing HEA-OCP. SA-FFCCPP [3] and GSOCCPP [4] produced the jitter of 8 and 10 ms, respectively. The jitter was reduced by 16% and 26% compared to SA-FFCCPP and GSOCCPP, respectively.

Table 6. Jitter.

Number of data packets	Jitter [ms]		
	HEA-OCP	SA-FFCCPP	GSOCCPP
25	6	8	10
50	8	10	12
75	10	12	14
100	11	13	15
125	12	14	16
150	13	16	18
175	15	17	20
200	17	19	21
225	18	20	22
250	19	22	24

5. Conclusion

An efficient controller placement problem for the distributed network was presented. This paper discussed a novel technique of HEA-OCP used for increasing the network of optimal controller placement. HEA-OCP uses the oppositional learned chaotic multi-criteria fish swarm optimization for solving the optimum CPP on multi-criteria functions, namely propagation latency, load balancing capability, bandwidth, fault tolerance, and data transmission rate. Simulations were conducted in the GBN network topology dataset with different optimization techniques. The performance assessment illustrates that the HEA-OCP technique is improved by a high packet delivery ratio, lesser latency, smaller jitter, and reduced packet drop rate.

References

1. V. Suma, Wearable IoT based distributed framework for ubiquitous computing, *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, **3**(01): 23–32, 2021, doi: 10.36548/jucct.2021.1.003.
2. J.V. Anand, Design and development of secure and sustainable software defined networks, *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, **1**(02): 110–120, 2019, doi: 10.36548/jucct.2019.2.005.
3. P. Aravind, G.P. Saradhi Varma, P.V.G.D. Prasad Reddy, Simulated annealing based optimal controller placement in software defined networks with capacity constraint and failure awareness, *Journal of King Saud University-Computer and Information Sciences*, **34**(8) Part B: 5721–5733, 2022, doi: 10.1016/j.jksuci.2021.04.012.
4. S. Torkamani-Azar, M. Jahanshahi, A new GSO based method for SDN controller placement, *Computer Communications*, **163**: 91–108, 2020, doi: 10.1016/j.comcom.2020.09.004.

5. G. Schütz, J.A. Martins, A comprehensive approach for optimizing controller placement in software-defined networks, *Computer Communications*, **159**: 198–205, 2020, doi: 10.1016/j.comcom.2020.05.008.

6. Y. Li, S. Guan, C. Zhang, W. Sun, Parameter optimization model of heuristic algorithms for controller placement problem in large-scale SDN, *IEEE Access*, **8**: 151668–151680, 2020, doi: 10.1109/ACCESS.2020.3017673.

7. A.K. Singh, S. Maurya, N. Kumar, S. Srivastava, Heuristic approaches for the reliable SDN controller placement problem, *Transaction on Emerging Telecommunication Technologies*, **31**(2): e3761, 2020, doi: 10.1002/ett.3761.

8. N. Samarji, M. Salamah, A fault tolerance metaheuristic-based scheme for controller placement problem in wireless software-defined networks, *Internationals Journal of Communication System*, **34**(4): e4624, 2021, doi: 10.1002/dac.4624.

9. B.R. Killi, S.V. Rao, Poly-stable matching based scalable controller placement with balancing constraints in SDN, *Computer Communications*, **154**: 82–91, 2020, doi: 10.1016/j.comcom.2020.02.053.

10. R. Soleymanifar, A. Srivastava, C. Beck, S. Salapaka, A clustering approach to edge controller placement in software-defined networks with cost balancing, *IFAC Papers OnLine*, **53**(2): 2642–2647, 2020, doi: 10.1016/j.ifacol.2020.12.379.

11. Y. Fan, L. Wang, X. Yuan, Controller placements for latency minimization of both primary and backup paths in SDN, *Computer Communications*, **163**: 35–50, 2020, doi: 10.1016/j.comcom.2020.09.001.

12. S. Tahmasebi, N. Rasouli, A.H. Kashefi, E. Rezabeyk, H.R. Faragardi, SYNCOP: An evolutionary multi-objective placement of SDN controllers for optimizing cost and network performance in WSNs, *Computer Networks*, **185**: 107727, 2021, doi: 10.1016/j.comnet.2020.107727.

13. R. Chai, Q. Yuan, L. Zhu, Q. Chen, Control plane delay minimization-based capacitated controller placement algorithm for SDN, *EURASIP Journal on Wireless Communications and Networking*, **2019**: 282, 17 pages, 2019, doi: 10.1186/s13638-019-1607-x.

14. N. Firouz, M. Masdari, A.B. Sangar, K. Majidzadeh, A novel controller placement algorithm based on network portioning concept and a hybrid discrete optimization algorithm for multi-controller software-defined networks, *Cluster Computing*, **24**: 2511–2544, 2021, doi: 10.1007/s10586-021-03264-w.

15. T. Hu *et al.*, An efficient approach to robust controller placement for link failures in software-defined networks, *Future Generation Computer Systems*, **124**: 187–205, 2021, doi: 10.1016/j.future.2021.05.022.

16. A.A. Ateya *et al.*, Chaotic salp swarm algorithm for SDN multi-controller networks, *Engineering Science and Technology, an International Journal*, **22**(4): 1001–1012, 2019, doi: 10.1016/j.jestch.2018.12.015.

17. A. Jalili, M. Keshtgari, A new reliable controller placement model for software-defined WANs, *Journal of AI and Data Mining*, **8**(2): 269–277, 2020, doi: 10.22044/jadm.2019.6319.1745.

18. Y.P. Llerena, P.R.L. Gondim, SDN-controller placement for D2D communications, *IEEE Access*, **7**: 169745–169761, 2019, doi: 10.1109/ACCESS.2019.2955434.

19. E. Tohidi, S. Parsaeefard, M.A. Maddah-Ali, B.H. Khalaj, A. Leon-Garcia, Near-optimal robust virtual controller placement in 5G software defined networks, *IEEE Transactions on Network Science and Engineering*, **8**(2): 1687–1697, 2021, doi: 10.1109/TNSE.2021.3068975.

20. A.K. Tran, M.J. Piran, C. Pham, SDN controller placement in IoT networks: An optimized submodularity-based approach, *Sensors*, **19**(24): 5474, 12 pages, 2019, doi: 10.3390/s19245474.

21. A. Dvir, Y. Haddad, A. Zilberman, The controller placement problem for wireless SDN, *Wireless Networks*, **25**: 4963–4978, 2019, doi: 10.1007/s11276-019-02077-5.

Received February 3, 2022; revised version August 13, 2022; accepted September 12, 2022.