http://dx.doi.org/10.21622/ACE.2024.04.2.852

A HYBRID-ABC APPROACH TO THE MULTI-CONTROLLER PLACEMENT PROBLEM OF SOFTWARE-DEFINED NETWORKS

Bilal Babayigit[1], Mohammed Abubaker[2], and Banu Ulu[3]

[a1] Computer Engineering Department, Faculty of Engineering, Erciyes University, Kayseri, Turkey
[b2] Computer Science Department, Palestine Technical College Deir El-Balah, Gaza, Palestine
[3] Software Engineering Department, Faculty of Engineering, Kayseri University, Kayseri, Turkey

Emails: {bilalb@erciyes.edu.tr, mabubaker@ptcdb.edu.ps, banuulu@kayseri.edu.tr}

Received on, 22 April 2024 - Accepted on, 29 July 2024 - Published on, 26 August 2024

ABSTRACT

The advent of the Software-Defined Network (SDN) paradigm brings a revolutionary approach to network management, improving agility and flexibility through centralized control of network nodes. Nevertheless, large-scale-SDN deployments introduce scalability issues, requiring multi-controller frameworks. This situation gives rise to the Controller Placement Problem (CPP), a challenging NP-hard problem that significantly impacts network performance. To tackle CPP, this paper introduces a hybrid-ABC algorithm tailored for multi-CPP contexts. The proposed algorithm combines the efficiency of the Artificial Bee Colony (ABC) meta-heuristic, known for rapidly reaching optimal solutions, with K-means clustering to improve solution quality and address the inherent exploitation deficiencies of ABC. The hybrid-ABC method thus integrates meta-heuristic swarm intelligence with machine learning techniques. The study evaluates the hybrid-ABC algorithm against the traditional ABC algorithm across two different datasets. Key performance metrics assessed include average latency times, solution quality, standard deviation, and running time. The experimental results demonstrate that the hybrid-ABC algorithm outperforms the traditional ABC algorithm in determining optimal controller placements within SDN architectures. For instance, in terms of average latency times, the hybrid-ABC achieved 1.33 ms compared to 2.88 ms for ABC on the first dataset, and 1.69 ms compared to 2.50 ms on the second dataset. Additionally, the hybrid-ABC algorithm showed superior solution quality (223.60 vs. 228.78) and lower standard deviation (2.98 vs. 4.31) with reduced running time (227.30 ms vs. 242.80 ms) for the first dataset. Similar trends were observed for the second dataset. These findings highlight the hybrid-ABC algorithm's superior performance in solving the multi-CPP problem, demonstrating its potential to enhance SDN network scalability and efficiency.

Index words: *ABC, controller placement problem, Hybrid-ABC, K-means, Multi-controller, Software-defined networking.*

I. INTRODUCTION

The rapid increase of Internet traffic brings challenges to traditional networking which has a complex and rigid structure, and this makes it difficult to manage and meet network requirements, especially in dynamic networks. Software-Define Network (SDN) offers network operators dynamic, flexible management, and rapid innovation. Therefore, in recent years SDN networks have become very attractive among researchers and different aspects of SDN are being addressed [1], [2]. To overcome the challenges of traditional network structures, SDN separates the data, application, and control layers from each other and allows direct programming of the network. The data plane is managed according to decisions made by the control plane. The application plane is the layer where applications such as big data analytics are built. The control plane is responsible for managing the traffic flow in the network and maintaining the connection between the data plane and the application plane. In particular, the controller of the control plane sets the configuration parameters, creates and makes the traffic forwarding rules, and manages the forwarding devices such as routers and switches. However, a single controller architecture of SDN control plane is inefficient to manage a large-scale network [3], [4]. Therefore, multi-controller designs are deployed in SDN networks to increase the reliability of the network, to improve its scalability, and avoid a single point of failure. However, in a multi-controller network architecture, the controller placement problem (CPP) is an important issue, as the location and the number of controllers have a large impact on network performance. CPP has three main concerns to be solved: optimal number of controllers, optimum location of the controllers, and the reduction of latency. Here, the latency is determined as the maximum value of the packet transmission latency between the switch and the controller in a high-performance environment. In the literature, there are several algorithms utilized for the CPP latency problem. These are K-means [5]–[7], particle swarm optimization algorithm [8], firefly algorithm [9], [10], and genetic algorithm [11]–[13], which are very important approaches in the field.

In these studies, the solution is to reduce the search space by network partitioning or to find a suitable result. However, considering the limited use of computing resources, research on efficient algorithms mainly focuses on search algorithms and aims to find approximately optimal solutions in a short time. In a recent study [4], the cost of controller setup is also evaluated. Therefore, closing a connection in low-traffic conditions or placing a controller in high-density cities are important issues to raise energy efficiency.

In this paper, ABC and Hybrid-ABC approaches are proposed to the CPP in SDN. It combines machine learning and swarm intelligence approaches to demonstrate their capability to achieve outstanding results in the area of CPP. This study aims to address the CPP in SDNs by introducing a hybrid-ABC algorithm specifically tailored for multi-CPP scenarios. The research seeks to explore whether integrating K-means clustering with the ABC algorithm can enhance solution quality over ABC algorithm for multi-CPP in SDNs. Two distinct datasets have been carefully created through the organization of data extracted from the ULAKNET database. The first dataset is created so that the positions of all cities relative to each other are minimum with the Dijkstra algorithm, while the second dataset is constructed by assigning weights to cities based on their population density ratios. Thus, this approach enables the efficient placement of controllers within high-density cities, thereby optimizing energy utilization. Both the ABC and the proposed hybrid-ABC algorithms have been applied to these datasets. The experimental findings demonstrate that the hybrid-ABC algorithm significantly reduces delay in solving the CPP problem, exhibiting superior performance in solution quality, standard deviation, and runtime when compared with the traditional ABC algorithm. Therefore, the proposed study represents a pioneering endeavor in the field of SDN by introducing the hybrid-

ABC algorithm, a solution specifically designed to tackle the multi-CPP problem. By integrating K-means clustering with ABC, the algorithm aims to overcome the limitations of traditional approach, offering superior performance in terms of solution quality, latency reduction, and execution time. This advancement holds the potential to revolutionize SDN network scalability and efficiency, paving the way for future innovations in network management.

The main contributions of this paper are as follows.

- *Energy-Efficient Controller Placement:* This work aims to conserve energy by strategically placing controllers in high-density cities, thereby addressing energy efficiency concerns within multi-CPP in SNDs.
- *Innovative ABC and Hybrid-ABC Approaches:* Exploring the application of the ABC algorithm to solve the multi-CPP in SNDs, while also proposing a new hybrid-ABC algorithm designed specifically to diminish latency within multi-CPP. The introduction and extensive utilization of the ABC and Hybrid-ABC methodologies represent pioneering endeavors focused on mitigating latency in multi-CPP. The proposed approach demonstrates significantly improved performance metrics, comprising heightened solution quality, reduced latency, and reduced execution time when compared to traditional ABC algorithms.

The rest of this paper is organized as follows: Section 2 presents the literature review. The methods are explained in Section 3. The experimental results and conclusions are provided in Sections 4 and 5, respectively.

II. LITERATURE REVIEW

The incorporation of multi-controller CPP in SDNs has attracted many researchers [14]–[17], enabling dynamic and adaptable network configurations and efficient resource allocation, thereby improving the reliability and scalability of the networks.

Lin et al. [18] propose a cost-effective controller placement scheme for software-defined vehicular networks (SDVN) to address the CPP in highly dynamic and complex networks. This reduced the number of controllers and improves their placement to decrease energy costs and enable green communication in SDVN. The proposed approach includes a minimum controller selection mechanism (MOSA) and an improved multi objective ABC algorithm by adding an adaptive fitness value mechanism based on bee behaviors to optimize controller placement and reduce the number of switched-on controllers. The performance is evaluated using conventional metrics such as delivery ratio, delay, and jitter. The results show that the proposed method can achieve a higher packet delivery ratio while reducing energy consumption and latency compared to other existing CPP schemes.

Wang et al. [5] introduce an approach to multi-CPP in SDN by utilizing network partition technique and an optimized K-means algorithm. The proposed approach focuses on minimizing latency between the controller and switches, which is a critical factor in SDN performance. The network is divided into subnetworks, and performance objectives can be implemented to the subnetworks instead of the whole network, which significantly reduces the complexity of the controller placement. The optimized K-means algorithm is used to divide the network and to minimize the maximum latency between the centroid and associated switches in subnetworks. The performance of the proposed optimized K-means algorithm is evaluated using two network topologies: the Internet2 OS3E topology and the Chinanet topology that is obtained from Topology Zoo. The evaluation focuses on latency performance and concludes that the optimized K-means algorithm outperformes the standard K-means algorithm in terms of the average maximum latency achieved.

Babayigit and Ulu [6] present a high available multi-controller structure for SDN and placement of multi-controllers of SDN with optimized K-means algorithm. The proposed system uses Docker-swarm mode to solve the single-point-of-failure problem and optimized K-means (Opk-means) algorithm to reduce the end-to-end latency. Experimental evaluations demonstrate that the proposed testbed effectively establishes a control plane for multi-controller environments, ensuring high availability. Furthermore, the Opk-means algorithm showcases a substantial reduction in latency compared to the standard K-means approach within the testbed, highlighting its efficiency in optimizing controller placement for superior performance in SDN networks.

Liao and Leung [8] propose a Multi-Objective Genetic Algorithm (MOGA) with a particle swarm optimization (PSO) based mutation function to solve the distributed CPP in SDNs. The proposed approach generates a pareto frontier with a larger diversity toward the given global best positions in much shorter convergence time than a general MOGA. The paper optimizes the three objectives which are minimizing switch-to-controller delay, minimizing controller-to-controller delay, and minimizing controller load imbalance. The PSO based mutation maintains a pre-calculated global best position for each single objective, and only uses a global best position to generate the velocity of a particle. The proposed approach is implemented in Matlab and uses an internet service provider network selected from Rocketfuel repository to evaluate its performance. The experimental results demonstrate the effectiveness of the proposed approach in solving the Distributed CPP in SDNs and outperforming the general MOGA in terms of convergence time and accuracy.

Zhang et al. [10] present an improved quantum-behavior particle swarm optimization algorithm called FE-QPSO for the CPP in SDNs. The proposed algorithm introduces two improved strategies, a full history elite strategy and a new dimension update strategy, to improve the shortcomings of traditional QPSO algorithms. The paper evaluates the performance of the proposed algorithm on the controller selection of three different types of classic topology models taken from the Topology Zoo website. The simulation results show that FE-QPSO achieves better performance than traditional QPSO algorithms in terms of the best controller's location of network topology and average latency.

Ahmadi and Khorramizadeh [12] propose an adaptive heuristic algorithm for multi-objective CPP in SDNs. A multi-start hybrid non-dominated sorting genetic algorithm is introduced to balance diversification and intensification in generating high-quality solutions. The algorithm commences by generating an initial population of switch numbers, then constructing a complete placement by situating controllers at various positions within the switches. The fitness function is defined as the sum of multiple objective functions, aiming to minimize the number of controllers, reduce the maximum load on switches, and minimize the maximum distance between switches and their respective controllers. A Pareto front is utilized to evaluate trade-offs among these objectives. The results, tested on several topologies from Internet Topology Zoo, demonstrate that the proposed algorithm requires less memory and computation time.

Radam et al. [19] introduce a simulated annealing for multi-controllers in SDN (SA-MCSDN) to solve the challenging multi-CPP in SDNs. Their proposed method utilizes an SA-based algorithm to effectively deploy controllers, thus reducing connection latency and propagation while improving throughput and reliability. The method involves generating the network, selecting an optimal controller using the firefly algorithm, and placing multiple controllers based on the selected controller using the SA-MCSDN algorithm. This approach considers the estimated distances and distribution times between the controllers and between controllers and switches to achieve the shortest distances and the least communication time. The authors simulate the proposed model using Network Simulator NS3 to extract the performance results. The SDN controller used in the simulation has 7 controllers, 28 switches, and 45 users. The simulation experiments demonstrate that the proposed SA-MCSDN provides a high-efficiency and scalable solution to optimize multi-CPP in SDNs.

In comparison to the existing works in the field of CPP within SDNs, the proposed hybrid-ABC algorithm stands out due to its unique integration of metaheuristic swarm intelligence and machine learning techniques. Hybrid methods integrating metaheuristics and machine learning constitute a novel research field. This innovative approach successfully combines machine learning with swarm intelligence methodologies and has demonstrated its ability to achieve outstanding results across various domains [20], [21]. This study seeks to prove its superiority in solving the multi-CPP problem in the SDN domain. While the mentioned several prior works have introduced various innovative algorithms and methodologies to address the challenges of multi-controller architectures and optimize controller placement, they primarily focus on specific optimization strategies such as simulated annealing, genetic algorithms, and particle swarm optimization. However, the distinctiveness of the hybrid-ABC algorithm lies in its fusion of the efficiency of ABC metaheuristic with K-means clustering, thereby offering a novel approach that leverages the strengths of both swarm intelligence and machine learning. This integration enables the proposed algorithm to enhance solution quality and also address the inherent limitations of traditional ABC algorithm, as demonstrated through comprehensive evaluations across diverse datasets. Consequently, the hybrid-ABC algorithm emerges as a promising solution for efficiently tackling the multi-CPP in SDNs, with its superior performance offering significant potential for enhancing the scalability and efficiency of SDN networks.

III. METHODS

A. THE CONTROLLER PLACEMENT PROBLEM

The controller placement problem (CPP) in SDN inherently tackles the question of how many controllers are needed and where to place them in the network. An SDN network is generally modeled as a $G(S; E; C)$ graph [10], with a set of switches (S), switch-to-switch or switch-to-controller connections (E), and controllers (C). In this modeling, it is assumed that each switch is connected to at least one controller. The shortest path cost between the controller and the switch can be represented as D_{cs}. The average and maximum latency between the switch and the controller are indicated as L_{asc} and L_{msc}, respectively, and the average and maximum latency between the controllers as L_{acc} and L_{mcc}, respectively. The four types of latency formulas are given by:

$$L_{asc} = \frac{1}{n}\sum d_{cs} \quad (1)$$

$$L_{msc} = \min d_{cs} \quad (2)$$

$$L_{acc} = \frac{1}{Q} \sum d_{ij} \tag{3}$$

$$L_{mcc} = \max d_{ij} \tag{4}$$

where the shortest path latency between each c_i and c_j controller pair are represented as d_{ij} in a D distance matrix and Q represents the total number of controller paths. The total average latency formula of SDN is obtained with the follows:

$$\min sum(L) = L_{asc} + L_{acc} \tag{5}$$

where, the objective function is to minimize the sum of the latency values, that is, the sum (L) value.

B. ARTIFICIAL BEE COLONY (ABC) ALGORITHM

The ABC algorithm [22] is an optimization technique based on the foraging mechanism of bees. One solution to the optimization problem is represented by the location of a food source. There are three different bee classes in the colony: worker, onlooker, and scout. Worker bees are responsible for sharing information with the bees that follow them and using the food source they identify. So, the number of population solutions is equal to the number of worker bees. Onlooker bees are responsible for exploiting the environment of the food source by selecting a food source from the population. Scout bees consist of worker bees whose solution has run out. The task of scout bees is to search for a new food source in the search space. The ABC algorithm starts with the generation of the solution population, as seen in Eq. (6).

$$x_{ij} = x^j_{min} + rand\ (0,1)(x^j_{max} - x^j_{min}) \tag{6}$$

where x^j_{min}, and x^j_{max} are the lower and upper bounds of the j^{th} parameter of the i^{th} solution. The fitness value of the algorithm is applied as

$$fit(x_i) = \begin{cases} \frac{1}{1+f_i(x_i)} \ if\ f_i(x_i) \geq 0 \\ 1 + abs(f_i\ (x_i))\ if\ f_i(x_i) < 0 \end{cases} \tag{7}$$

where $f_i\ (x_i)$ is the objective function value of the i^{th} solution x_i.

Then, in the algorithm, the worker bees are sent to food sources to explore neighborhoods looking for better resources using Eq. (8).

$$v_{ij} = x_{ij} + \emptyset_{ij}\ (x_{ij} - x_{kj}) \tag{8}$$

In Eq. (8), $\emptyset_{ij}$ is a random number belonging to the range [-1,1], x_k is a food source position chosen randomly so that $k \neq i$ and j is a randomly selected dimension.

Onlooker bee applies a fitness-based selection among solutions of the population defined as

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{9}$$

Scout bees identify the exhausted solutions and replace the exhausted solutions with randomly generated new solutions using Eq. (6). Finally, the best solution is stored in memory and the process given in Eqs. (6-9) is repeated until the completion condition is met.

C. HYBRID-ABC ALGORITHM

Although the ABC algorithm has few parameters and gives fairly good results in many problems such as clustering, ABC has the significant disadvantage of poor exploitation of solutions resulting from the randomness of the search process of neighboring solutions. To eliminate this disadvantage, ABC can be hybridized with other algorithms. In this paper, a Hybrid-ABC which is a hybridization of K-means and ABC is presented to obtain better solutions.

K-means is a well-known clustering algorithm that aims to divide data into K clusters based on similarity [23], [24]. By efficiently allocating data points to the closest cluster centroid, it reduces the inter-cluster variation. Therefore, the motivation behind using the K-means in the context of ABC can help in refining the solutions obtained through the random search process. In which, K-means can refine and improve the initial solutions by assigning them to the nearest cluster centroid, leading to a better exploration of the search space. Also, by integrating K-means within the ABC, the hybrid algorithm could benefit from the local search capabilities of K-means to fine-tune and improve the solutions found by ABC. Consequently, the proposed Hybrid-ABC approach can mitigate the weakness related to poor exploitation of solutions of ABC by leveraging the clustering capabilities of K-means for refining and improving the solutions obtained through the random search process, thus enhancing the overall performance in optimization problem of solving the multi-CPP within SDN architectures.

The Hybrid-ABC algorithm has two fundamentally different features from the classical ABC algorithm. The first of these; the gbest value is determined by the K-means algorithm before the ABC algorithm is run. So, the ABC algorithm is initialized with this gbest value. Secondly, two different approaches are proposed to calculate the neighborhood of worker bees and onlooker bees as Eq. (10) for worker bees and Eq. (11) for onlooker bees.

$$v_i = x_i(g) + k_i \times \left(x_{i1}(g) - x_i(g)\right) + F' \times \left(x_{i2}(g) - x_{i3}(g)\right) \qquad (10)$$

where i_1, i_2 and i_3 are indexes of food source randomly chosen so that $i_1 \neq i_2 \neq i_3 \neq i$. k_{ij} is a random value from a uniform distribution between 0 and 1. $F' \in [0,1]$ a random value from a uniform distribution created once for each iteration.

$$v_{ij} = gbest_j + F \times (x_{dets,j} - x_{src,j}) \qquad (11)$$

where $gbest$ denotes the bee with the currently best fitness value. $x_{dest,j}$ and $x_{src,j}$ are the bees chosen randomly such that $f(x_{dest,j}) < f(x_{src,j})$, f being the objective function to minimize. F here is a randomly chosen value from a uniform distribution between 0 and 1. The steps of the hybrid ABC algorithm are given in Algorithm 1.

Algorithm 1 Hybrid-ABC algorithm.

```
1:  Initialize the population of solutions $x_{ij}$;
2:  Initialize gbest using K-means algorithm;
3:  Evaluate the fitness value $fit(x_i)$ of solutions $x_i$ using Eq. 7;
4:  **repeat**
5:  Produce a new solution using Eq.10;
6:  Evaluate the fitness value of the new solution using Eq. 7;
7:  Calculate the probabilities using Eq.9;
8:  Produce the new solutions (new positions) $v_i$ for the onlookers from the solutions $x_i$, selected depending on $P_i$, and evaluate them;
9:  Determine the abandoned solution (source), if exists, and replace it with a new randomly produced solution $x_i$ for the scout using the Eq. 6;
10: Memorize the best solution found so far;
11: **until** stop condition is satisfied
12: **return** $gbest$;
```

IV. EXPERIMENTAL RESULTS

In this paper, all simulation experiments are performed on Windows 10 operating system in a computer with an Intel i7 processor and 8 GB of RAM. ABC and the proposed Hybrid-ABC codes are written in Python and run on the PyCharm platform. For both ABC and Hybrid-ABC algorithms, the following parameters are used. The population number, the limit value, and the maximum number of iterations are set as 100, 50, and 50, respectively.

In the simulations, two different datasets are used. The First Dataset is prepared using the ULAKNET dataset from the Topology Zoo website, while the Second Dataset is created by weighting each node of the data in the First Dataset according to their population density ratio. In the generation of datasets, connections between cities (nodes) are used. The number of nodes between both cities is determined to be the shortest by using the Dijkstra algorithm. Determined node values are kept in a matrix in the form of 81 × 81. This matrix is used as the First Dataset. Having a controller in the regions, where the network density is higher in SDN, reduces the delay in the network and increases the efficiency of the network. For this reason, the Second Dataset is created as each node is weighted according to population density. The weighting formula for each node is given as follows,

$$^{p}/_{tp} \times n = new\_n \tag{12}$$

where p, tp, n, new_n values represent the population of a city, the total population, the node, and the newly produced node value, respectively. Thus, the First Dataset is primarily focused on network topology derived from the ULAKNET dataset, while the Second Dataset emphasizes population density as a criterion to weight nodes for optimizing SDN. Table 1 demonstrates the main characteristics of the used datasets.

The created datasets are given as inputs to the algorithms. The results obtained from the algorithms are repeated 30 times and the most frequently repeated controller locations are determined. The controller placements obtained for the first run with the first and second datasets as input for ABC and Hybrid-ABC algorithms are shown in Figures 1 and 2, respectively. Besides that, the most frequently repeated controller placements are given in Figures 3 and 4, with the First Dataset and the Second Dataset as input , for ABC and Hybrid-ABC algorithms. In Figures 3 and 4, the orange nodes denote the switches, and the red nodes denote the controllers.

TABLE I
DATASETS CHARACTERISTICS

Datasets	Characteristics
The First Dataset	- Derived from ULAKNET dataset from the Topology Zoo website. - Generated using connections between cities (nodes). - Utilizes the Dijkstra algorithm to determine shortest paths between cities. - Stored as a matrix in an 81 × 81 format.
The Second Dataset	- Shared the same characteristics of the First Dataset. - Created by weighting nodes based on population density ratio. - Intended to optimize SDN by assigning weights to each node.

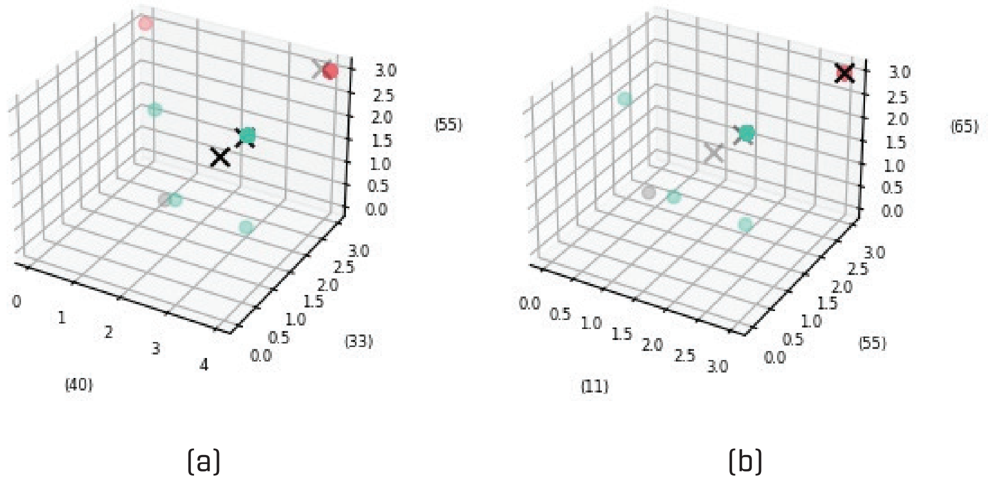


Fig. 1. Controller placements result for the First Dataset as input after one iteration run according to: (a) ABC, (b) Hybrid-ABC

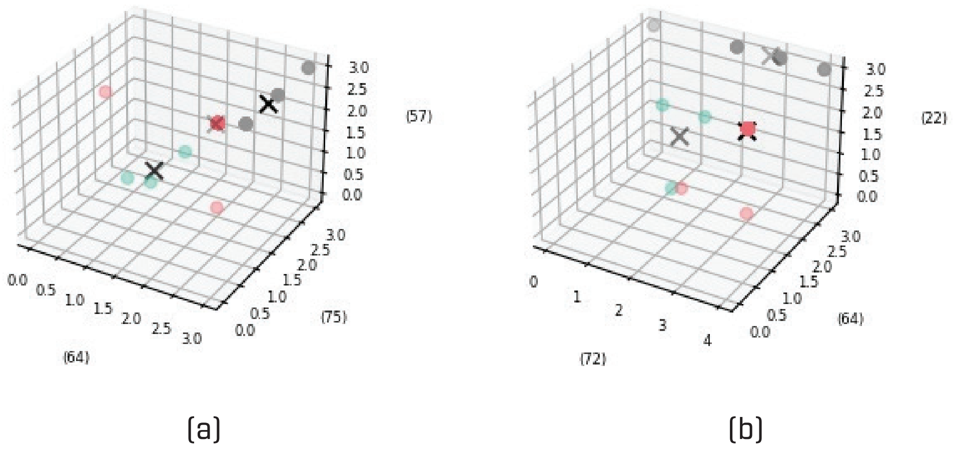
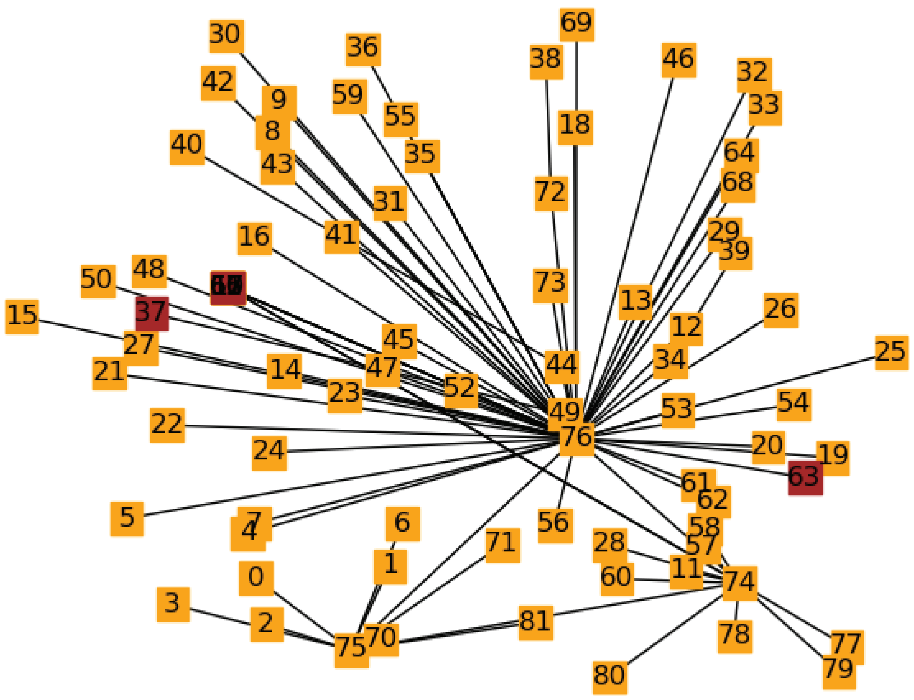
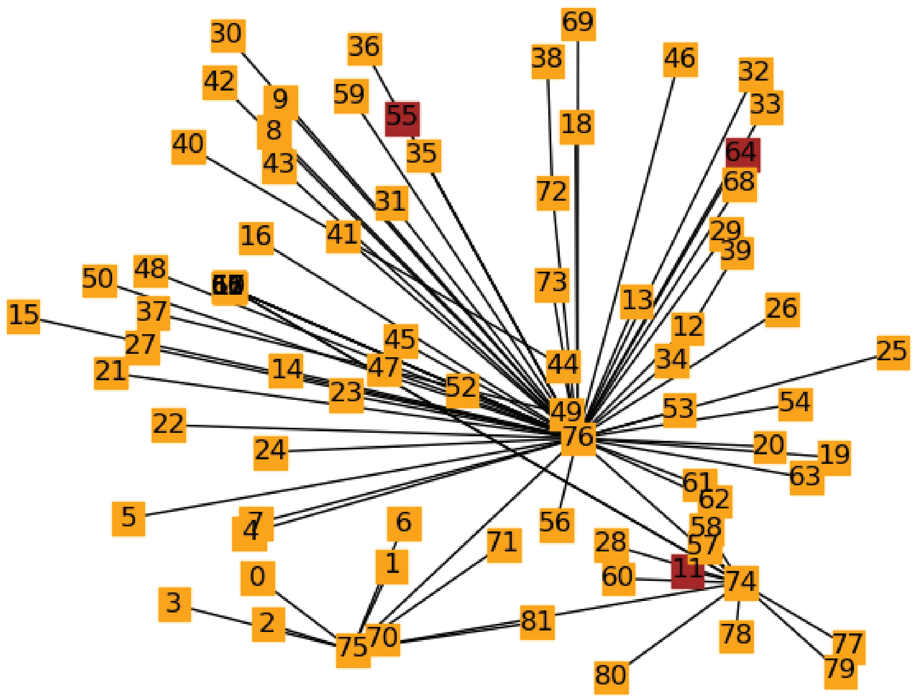


Fig. 2. Controller placements result for the Second Dataset as input after one iteration run according to: (a) ABC, (b) Hybrid-ABC

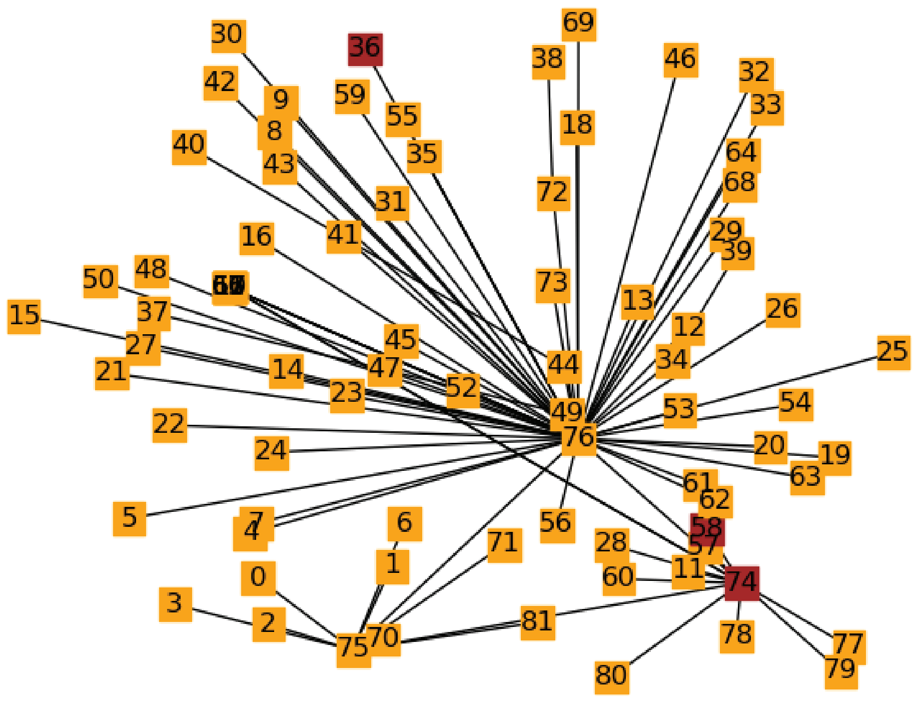


(a)

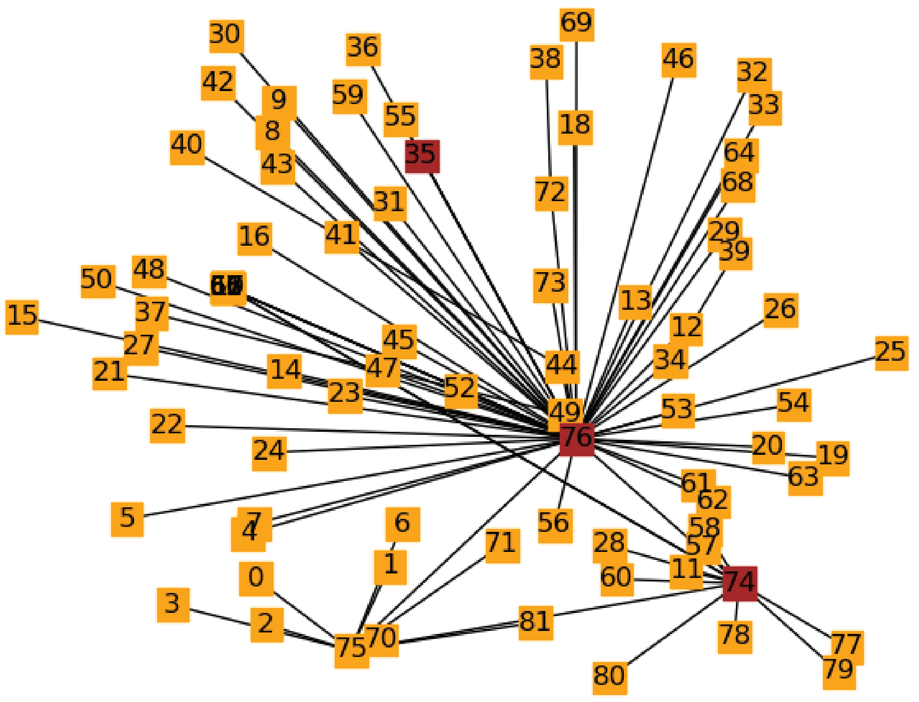


(b)

Fig. 3. Most repeated controller placements result for the First Dataset as input after one iteration run according to: (a) ABC, (b) Hybrid-ABC



(a)



(b)

Fig. 4. Most repeated controller placements result for the Second Dataset as input after one iteration run according to: (a) ABC, (b) Hybrid-ABC

In Figures 3 and 4, 76 represents the first-busiest province, while 74 represents the second-busiest province of Turkey. The controller placement directly affects network performance in SDN; therefore, in Fig. 3, when the results obtained from the First Dataset are compared with the results obtained from the Second Dataset in Fig. 4, it is obvious that the controller placement in the Second Dataset is more efficient in terms of energy-efficiency as the Second Dataset is characterized the population density of each city.

In the controller placement problem, it is aimed to minimize the average delay according to the controller positions and, find approximately optimal solutions in a short time. In Table II, the average latency times obtained as a result of placing the controllers according to both datasets are given milliseconds and the results of ABC and Hybrid-ABC algorithms in terms of quality of solution, standard deviation, and running time are listed in Table III for the first and the second dataset.

TABLE II
COMPARISON OF THE AVERAGE LATENCY TIMES OF THE CONTROLLERS FOR ABC AND HYBRID-ABC ALGORITHMS

Datasets	Algorithms	Controllers Average Latency (ms)
First Dataset	ABC	2.88
	Hybrid-ABC	1.33
Second Dataset	ABC	2.50
	Hybrid-ABC	1.69

In Table II, the controller latency times of ABC and Hybrid-ABC algorithms for two different datasets are compared.

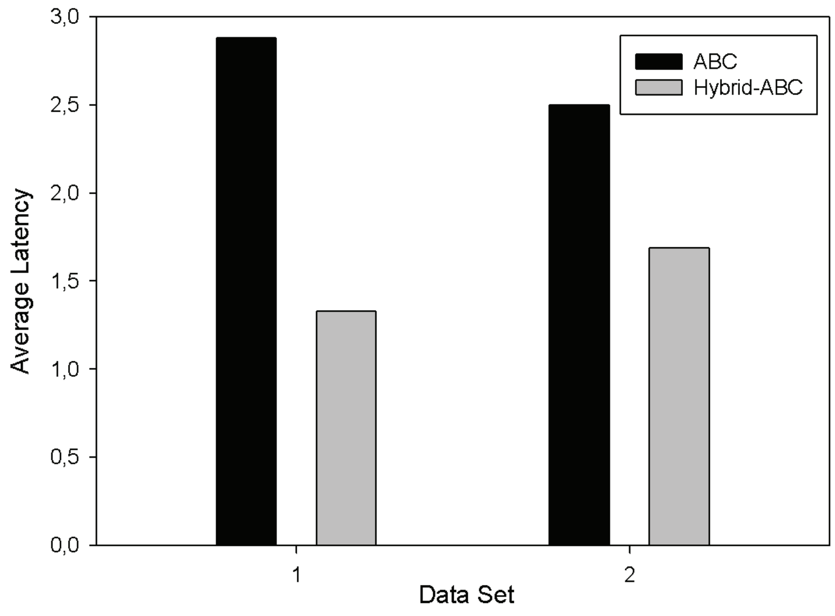


Fig. 5. Average latency results of ABC and Hybrid-ABC for both datasets

It is clearly seen from Table II and Fig. 5 that the Hybrid-ABC algorithm finds a better location than the ABC algorithm and reduces the latency. This comparison between ABC and Hybrid-ABC algorithms for addressing the multi-CPP in SDN across two datasets, reveals significant improvements in average latency. The Hybrid-ABC algorithm outperforms the traditional ABC approach by reducing latency by approximately 53.82% (2.88 ms to 1.33 ms) and 32.40% (2.50 ms to 1.69 ms) across the First and the Second datasets, respectively. These results indicate that the integration of K-means within the Hybrid-ABC algorithm effectively optimizes controller placement due to the fact that K-means clustering facilitates more effective grouping of network elements which aids Hybrid-ABC in making informed decisions about controller placement based on network characteristics. Consequently, this enabled the proposed algorithm to minimize communication latency between controllers and switches by strategically positioning them.

TABLE III
COMPARISON PERFORMANCE OFABC AND HYBRID-ABC ALGORITHMS FOR THE CPP PROBLEM

Datasets	Algorithms	Quality of Solution	Standard Deviation	Running Time (ms)
First Dataset	ABC	228.78	4.31	242.80
	Hybrid-ABC	223.60	2.98	227.30
Second Dataset	ABC	256.92	4.24	234.98
	Hybrid-ABC	254.96	0.95	233.95

In Table III, the solution quality is measured by the sum of squares error. Table III clearly shows that the performance of the proposed Hybrid-ABC algorithm is better than that of the ABC algorithm. It is observed from Table III that with the use of Hybrid-ABC, the standard deviation is decreased for datasets and the quality of the solutions is augmented. In addition, K-means has a considerable impact on the convergence of the algorithm in the use of Hybrid-ABC, therefore, the running time is decreased for datasets. The enhancements in quality of solution, standard deviation, and running time can be attributed to the integration of K-means clustering with the ABC algorithm. This combination improves the exploitation capabilities of the ABC algorithm, allowing for more efficient and accurate controller placements. For instance, in the first dataset, the Hybrid-ABC algorithm achieves a quality of solution of 223.60 compared to 228.78 with the traditional ABC, a reduction in standard deviation from 4.31 to 2.98, and a decrease in running time from 242.80 ms to 227.30 ms. In the second dataset, the Hybrid-ABC algorithm achieves a quality of solution of 254.96 compared to 256.92 with the traditional ABC, a reduction in standard deviation from 4.24 to 0.95, and a decrease in running time from 234.98 ms to 233.95 ms. The K-means clustering component helps in better initialization and refinement of solutions, leading to a higher quality of solution with lower standard deviation. This reduces the variability in the results and ensures more consistent performance. Additionally, the refined search process results in faster convergence, thereby reducing the running time. However, it is important to note that the integration of K-means clustering may introduce some constraints, such as potential sensitivity to the initial cluster centers and outliers. These constraints will be considered and addressed in future work to further enhance the robustness and efficiency of the Hybrid-ABC algorithm.

Consequently, energy efficiency in SDN is closely related to the optimal placement of controllers. Proper controller placement minimizes the distance between controllers and network nodes, which in turn reduces the amount of data that needs to be transmitted over the network. This reduction in data transmission can lead to lower energy consumption by reducing the load on network links and switches. The researchers' proposed Hybrid-ABC algorithm improves solution quality and reduces latency, as evidenced by the results in Tables II and III. Lower latency and better-quality solutions not only enhance network performance but also contribute to energy efficiency. By optimizing controller placement, the researchers' approach helps in reducing the overall energy required for data transmission and processing. Thus, the benefits of lower latency and improved solution quality indirectly support better energy efficiency.

V. CONCLUSION

In this paper, the researchers' introduced ABC and proposed Hybrid-ABC algorithms for solving the multi-CPP within SDN environments. The proposed Hybrid-ABC algorithm combines machine learning and swarm intelligence approaches, showcasing its potential to deliver successful results in solving the CPP. To evaluate these algorithms, the researchers' constructed two distinct datasets. The initial dataset is formulated by rearranging the ULAKNET data sourced from the Topology Zoo database. Subsequently, we generated a second dataset by applying weights based on population density to the initial data, thereby enhancing energy efficiency within the controller setup and layout. The proposed Hybrid-ABC algorithm is devised by leveraging the K-means algorithm to determine the $gbest$ value, along with integrating diverse neighborhood development strategies into the traditional ABC approach. The researchers' experimental results demonstrated that the proposed Hybrid-ABC approach can reduce controller latency and algorithm running time across both datasets. Therefore, the proposed Hybrid-ABC algorithm exhibits promising potential in enhancing network performance metrics, thereby emphasizing its crucial significance in advancing the efficacy and efficiency of SDNs. Consequently, by enhancing the performance of SDNs, organizations and network operators can access numerous benefits, including increased agility, scalability, cost-effectiveness, adaptability to new technologies, and overall network robustness, which are vital in dynamic and continually evolving networking environments. As a direction for future research, the researchers aim to delve deeper into exploring simpler yet more effective optimization algorithms relevant to the multi-CPP. Additionally, comparative studies between these algorithms and the Hybrid-ABC approach will be conducted to evaluate their respective efficiencies and performance in the CPP within SDN environments.

VI. REFERENCES

[1] R. Amin, E. Rojas, A. Aqdus, S. Ramzan, D. Casillas-Perez, and J. M. Arco, "A Survey on Machine Learning Techniques for Routing Optimization in SDN," 2021. doi: 10.1109/ACCESS.2021.3099092.

[2] D. Espinel Sarmiento, A. Lebre, L. Nussbaum, and A. Chari, "Decentralized SDN Control Plane for a Distributed Cloud-Edge Infrastructure: A Survey," IEEE Communications Surveys and Tutorials, vol. 23, no. 1, 2021, doi: 10.1109/COMST.2021.3050297.

[3] T. Hu, Z. Guo, P. Yi, T. Baker, and J. Lan, "Multi-controller Based Software-Defined Networking: A Survey," IEEE Access, vol. 6, 2018, doi: 10.1109/ACCESS.2018.2814738.

[4] J. Lu, Z. Zhang, T. Hu, P. Yi, and J. Lan, "A survey of controller placement problem in software-defined networking," IEEE Access, vol. 7, 2019, doi: 10.1109/ACCESS.2019.2893283.

[5] G. Wang, Y. Zhao, J. Huang, Q. Duan, and J. Li, "A K-means-based network partition algorithm for controller placement in software defined network," in 2016 IEEE International Conference on Communications, ICC 2016, 2016. doi: 10.1109/ICC.2016.7511441.

[6] B. BABAYİĞİT and B. ULU, "A High Available Multi-Controller Structure for SDN and Placement of Multi-Controllers of SDN with Optimized K-means Algorithm," Iğdır *Ü*niversitesi Fen Bilimleri Enstitüsü Dergisi, vol. 11, no. 4, 2021, doi: 10.21597/jist.932575.

[7] B. P. R. Killi, E. A. Reddy, and S. V. Rao, "Cooperative game theory based network partitioning for controller placement in SDN," in 2018 10th International Conference on Communication Systems and Networks, COMSNETS 2018, 2018. doi: 10.1109/COMSNETS.2018.8328186.

[8] L. Liao and V. C. M. Leung, "Genetic algorithms with particle swarm optimization based mutation for distributed controller placement in SDNs," in 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2017, 2017. doi: 10.1109/NFV-SDN.2017.8169836.

[9] K. S. Sahoo, S. Sahoo, A. Sarkar, B. Sahoo, and R. Dash, "On the placement of controllers for designing a wide area software defined networks," in IEEE Region 10 Annual International Conference, Proceedings/TENCON, 2017. doi: 10.1109/TENCON.2017.8228398.

[10] Q. Zhang et al., "A new quantum particle swarm optimization algorithm for controller placement problem in software-defined networking," Computers and Electrical Engineering, vol. 95, 2021, doi: 10.1016/j.compeleceng.2021.107456.

[11] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," IEEE Communications Letters, vol. 21, no. 4, 2017, doi: 10.1109/LCOMM.2016.2645558.

[12] V. Ahmadi and M. Khorramizadeh, "An adaptive heuristic for multi-objective controller placement in software-defined networks," Computers and Electrical Engineering, vol. 66, 2018, doi: 10.1016/j.compeleceng.2017.12.043.

[13] B. Babayigit, B. Ulu, and E. N. Hascokadar, "Solving Multi-Controller Placement Problem in Software Defined Networks with A Genetic Algorithm," in UBMK 2019 - Proceedings, 4th International Conference on Computer Science and Engineering, 2019. doi: 10.1109/UBMK.2019.8907199.

[14] G. D'Angelo and F. Palmieri, "A co-evolutionary genetic algorithm for robust and balanced controller placement in software-defined networks," Journal of Network and Computer Applications, vol. 212, 2023, doi: 10.1016/j.jnca.2023.103583.

http://dx.doi.org/10.21622/ACE.2024.04.2.852

[15] D. He, J. Chen, and X. Qiu, "A density algorithm for controller placement problem in software defined wide area networks," Journal of Supercomputing, vol. 79, no. 5, 2023, doi: 10.1007/s11227-022-04873-x.

[16] M. Dhar, A. Debnath, B. K. Bhattacharyya, M. K. Debbarma, and S. Debbarma, "A comprehensive study of different objectives and solutions of controller placement problem in software-defined networks," Transactions on Emerging Telecommunications Technologies, vol. 33, no. 5, 2022, doi: 10.1002/ett.4440.

[17] S. Guan, J. Li, Y. Li, and Z. Wang, "A multi-controller placement method for software defined network based on improved firefly algorithm," Transactions on Emerging Telecommunications Technologies, vol. 33, no. 7, 2022, doi: 10.1002/ett.4482.

[18] N. Lin, Q. Zhao, L. Zhao, A. Hawbani, L. Liu, and G. Min, "A Novel Cost-Effective Controller Placement Scheme for Software-Defined Vehicular Networks," IEEE Internet Things J, vol. 8, no. 18, 2021, doi: 10.1109/JIOT.2021.3069878.

[19] N. S. Radam, S. T. F. Al-Janabi, and K. S. Jasim, "Using Metaheuristics (SA-MCSDN) Optimized for Multi-Controller Placement in Software-Defined Networking," Future Internet, vol. 15, no. 1, 2023, doi: 10.3390/fi15010039.

[20] S. Malakar, M. Ghosh, S. Bhowmik, R. Sarkar, and M. Nasipuri, "A GA based hierarchical feature selection approach for handwritten word recognition," Neural Comput Appl, vol. 32, no. 7, 2020, doi: 10.1007/s00521-018-3937-8.

[21] N. Bacanin, R. Stoean, M. Zivkovic, A. Petrovic, T. A. Rashid, and T. Bezdan, "Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization," Mathematics, vol. 9, no. 21, 2021, doi: 10.3390/math9212705.

[22] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," Journal of Global Optimization, vol. 39, no. 3, 2007, doi: 10.1007/s10898-007-9149-x.

[23] M. Abubaker and W. Ashour, "Efficient Data Clustering Algorithms: Improvements over Kmeans," International Journal of Intelligent Systems and Applications, vol. 5, no. 3, 2013, doi: 10.5815/ijisa.2013.03.04.

[24] M. B. A. - and H. M. H. -, "Kmeans-Based Convex Hull Triangulation Clustering Algorithm," Research Notes in Information Science, vol. 9, no. 1, 2012, doi: 10.4156/rnis.vol9.issue1.3.