# WOACCPP: Wisdom Of Artificial Crowds for Controller Placement Problem with Latency and Reliability in SDN-WAN

**Vidya Sagar Thalapala** ( haividya@gmail.com )
Indian Institute of Information Technology Kottayam
**Anandhu Mohan**
Indian Institute of Information Technology Kottayam
**Koppala Guravaiah**
Indian Institute of Information Technology Kottayam

**Research Article**

**Keywords:** Software Defined Networking (SDN), Controller Placement Problem (CPP), Genetic Algorithm, Network Latency, Reliability

**Posted Date:** May 26th, 2022

**DOI:** https://doi.org/10.21203/rs.3.rs-1660537/v1

# WOACCPP: Wisdom Of Artificial Crowds for Controller Placement Problem with Latency and Reliability in SDN-WAN

Vidya Sagar Thalapala[1*], Anandhu Mohan[2†] and Guravaiah Koppala[1†]

[1*]Department of Computer Science & Engineering.
[2]Department of Computational Science & Humanities.
[1,2*] Indian Institute of Information Technology Kottayam, 683365, Kerala, India.

*Corresponding author(s). E-mail(s): haividya@gmail.com;
Contributing authors: anandhum.phd203002@iiitkottayam.ac.in;
kguravaiah@iiitkottayam.ac.in;
†These authors contributed equally to this work.

**Abstract**

The highly motivated Software-Defined Networking (SDN), the network programmability architecture made network maintenance easy. Multiple controllers are used to provide better scalability and reliability to the network in SDN. However, to enhance the network's performance in networks at a large scale, the placement of controllers needs to be optimized. The Wisdom of Artificial Crowds (WOAC) genetic algorithm is used to minimize propagation latency by using the fewest possible controllers. To evaluate the performance of proposed technique the Internet2 OS3E topology is considered. The proposed controller placement technique reduces latency and improves reliability by comparing the existing solutions namely Modified Density Peak Clustering (MDPC) and optimized K-means approaches. The proposed method out performs existing techniques by 10% improves in latency and maximizes the reliability.

**Keywords:** Software Defined Networking (SDN), Controller Placement Problem (CPP), Genetic Algorithm, Network Latency, Reliability

# 1 Introduction

The networking business is at a crossroads right now [1]. Its because of the rapid growth of communications technology and the increase in internet users daily. In the same way, the network enactment needs to improve according to the demand-supply strategies, and resource deployment must avoid service disruption, and congestion [2]. So, it requires an extensive range of services that demands network reliability and scalability, which traditional networks cannot provide. As a result, Software-Defined Networking (SDN) was developed [3]. The SDN is distinguished by separating the associated control and forwarding functionalities. The functionality of the controller improves the network's dependability and robustness. Compared to traditional network topologies, these qualities of SDN make it an excellent choice for delivering QoS for applications in more manageable and more adaptable ways [4]. By adopting SDN, software developers can easily control and manage network resources with well defined programming [5]. In SDN, Packets are dealt with forwarding tables by SDN switches. SDN switches can act as a router, switch, NAT, firewall, and so on in terms of the policies set by the centralized controller on forwarding tables. The separation of the control plane from the data plane simplifies network management and allows for more innovation [6]. A single controller is adequate for a small-scale SDN. The single controller can increase the delay between the switch and controller when the network scale grows. In addition, the controller may collapse, and the entire network will be rendered useless. The increase in the size of a network also requires more controllers to handle large-scale SDN by considering the parameters such as latency, performance, security, and reliability [7–10].

In a multi-domain environment, each domain contains a controller, and it is responsible for routing packets inside its domain independently. As a result, it ignores (1) routing decisions in other domains, (2) the current state of other domains in terms of utilization and congestion, and (3) overall application performance requirements (e.g., bandwidth and end-to-end delay of flows associated with applications), and (4) monetary costs, i.e., the cost incurred when a flow uses one or more domains. As a consequence, it's hard to ensure their performance when network applications associated traffic flows across various administrative domains [11]. Thus, the switch-controller connection delay affects the flow setup time in SDN [12]. Thus, careful planning is required to define the best location for the controller to provide adequate performance and a stable SDN. When deploying multiple controllers, the deployment of controllers depends on numerous metrics such as scalability, fault tolerance and reliability are also taken into account. Due to the dynamic network traffic, controllers' load balancing and efficiency are also managed. The dynamic allocation multi-controllers reduce the controller's response time and optimize the communication overhead between controllers and switches [13]. Furthermore, a suitable decision must be taken to determine the number of controllers 1) that cover the complete network and 2) the number of switch nodes nailed by the controllers more reliably.

Nowadays, achieving security and networking reliability is a significant priority due to faults or failures in Network.The fault tolerance of the data planes and the control planes determine the SDN environment's reliability [14]. The network's reliability mainly depends on two factors, i.e., node failure and link failure. Link failure is a frequently appearing problem in the network. The communication disconnects the controller and its corresponding switches whenever the connection fails. In the meantime, the switch drops the packets that do not have the entry in the flow table before reconnecting the backup path. Due to link failure, latency may increase, necessitating careful planning.

Heller and Sherwood [15] began their investigation by looking into the controller placement problem (CPP). They examined how placement affected average latency, and worst-case latency in the Internet2 OS3E topology [16]. The robustness of CPP also depends on latency and other metrics such as reliability and load balancing. The Controller Placement Problem (CPP) is an optimization problem comparable to the facility location problem, classified as NP-hard [15].

Exact solutions to NP-hard issues are thought to take exponential time [17]. Because solving such issues with traditional computer architectures is impractical, optimal approaches are substituted with approximation algorithms, typically taking polynomial time to yield adequate results [18].

Meta-heuristics are heuristic algorithms that can solve a variety of issues. These algorithms are computational approaches that iteratively improve a candidate's response against a set of quality criteria to get a close approximation to an optimal solution [19]. The majority of meta-heuristic optimization and search algorithms are based on biological processes such as genetic algorithms (GAs) [20].

The Wisdom of Artificial Crowds (WOAC) technique is intended to be used as a post-processing step for any swarm-based optimization approach that generates a population of intermediate solutions [19]. A crowd of intelligent agents represents the population of intermediate solutions to a problem. An aggregation approach is devised for a specific problem that permits individual solutions from the population to produce a higher one. This aggregated population motivates the authors to implement the proposed system by considering the WOAC approach.

The following are the paper's primary contributions in summary.

- Network segmentation and controller placement are recommended to increase the performance of SDN networks and controller scalability. Therefore, the proposed strategy's primary performance objectives were implemented on sub-networks and significantly decreased controller placement computations.
- The network partition problem is analytically composed, and the controller placement problem is defined in terms of latency and reliability.
- Proposed Wisdom of Artificial Crowds based Controller Placement Problem (WOACCPP) for better latency and reliability.

- The suggested WOACCPP outshines other methods such as MDPC and the optimized K-means algorithm for optimal latency and network reliability.

The remaining paper explains the following: Related work for the control placement problem is presented in Section 2. Section 3 listed the mathematical representation of the network graph and the WOACCPP approach for CPP Section 4. Finally, section 5 explains the performance evaluation, followed by a conclusion.

## 2 Related Work

In recent years researchers proposed different approaches to solve the controller placement problem to reduce latency and improves the reliability, which in turn improves the scalability. Some of the relevant approaches are discussed in this section.

Heller et al. [15] initially presented the controller placement problem in SDN, which is a K-center (K-median) problem. They mainly examined placement effects on propagation delays and established several measures that formed the groundwork for future research on the subject.

Guang et al. [21] employed the capacitated K-Center issue to find the radius with the minimum propagation delay, as well as dynamic load balancing of the controllers based on their capacity. According to simulation results, their method can reduce the number of controllers and the strain on the busiest traffic controllers. An optimized K-means technique is proposed to make a hefty latency as low as possible between the centroid and associated switches in sub-partition of the network [22]. Compared to standard K-means, optimized K-means may ensure that each partition lowers the maximum delay between the centroid and other nodes, resulting in dramatically enhanced performance.

Zhong et al. [23] uses the min-cut problem to identify the reliable controller position with minimal possibility of a delay between the controller and their connected switches. Liao et al. [24] use the Density-Based controller placement technique to minimize the number of controllers in the SDN network. In the first step of the approach, authors created a switch table with densities and their related information. Using the above table data, the CPP was then resolved using a density-based clustering approach.

To reduce frequent administrative interruption, latency, and disconnections, B P Rao et al. [25] developed a controller placement approach that considers controller dependability, capacity, and contingency preparations in the case of a controller failure. The goal is to reduce the time it takes to move to the nearest centroid while still retaining enough capacity and latency. They also created a more comprehensive model to reduce average latency while accounting for various controller failures. The genetic algorithm NAGA II [26] is utilized to accommodate many objectives and limitations such as scalability, reliability, etc. In addition, the authors developed a theoretical approach for addressing a node cluster configuration that maximizes the sum of clusters' average edge connectivity while minimizing cluster imbalance.

Using Particle Swarm Optimization (PSO) and FireFly (FFA), K S Sahoo et al. [27] developed two population-based meta-heuristic approaches to solve the CPP on a set of realistic topologies. The Cluster Leader Election graph optimization problem is used in the literature [28] to reduce the distances between the switch and controller nodes. Furthermore, while dealing with a set cover problem, the greedy method produces greater interactability and approximation.

The Clustering-based Network Partition Algorithm [29] is a method for minimizing end-to-end latency, which includes processing, propagation, and transmission. This research looks at queuing latency, how to reduce it using the queuing theorem, and how to put controllers in place for each subnetwork to achieve total network latency. Chai et al. [30] presented a heuristic approach based on the Dijkstra and K-Means algorithms to reduce network latency. They also used the Kuhn-Munkres (K-M) algorithm to determine the best controller placement.

The Knapsack 0-1 issue is used by the authors in literature [31] to investigate controllers' processing rates and costs. They also describe a new technique for decreasing the propagation delay on capacitated controllers while keeping the execution time as fast as possible. The Iterated Local Search (ILS) technique addresses the controller placement problem [32]. The neighbourhood and perturbation mechanisms are used in this technique to create a network topology that is both efficient and speedy.

CPP's mathematical functions are ideally suited to the shortest possible execution time. To generate a distance matrix and compute bounded distances, Dhar et al. [33] employed the mathematical function "$-method" to reduce average switch-to-controller latency. The Joint Delay and Reliability aware Controller Placement Problem [34] is a new measure that accounts for both the differences between the controller and its attached switches as well as the inter-controller latency. It lowers the cumulative latency of the entire network.

Yuezhen et al. [35] present a network partitioning technique based on a modified density peaks clustering (MDPC) algorithm that clusters the switches to divide the network into many sub-networks. The method assesses the switches' average degree and nearness centrality to the controller with minimum average latency.

Kanodia et al. [36] used Grey Wolf Optimization(GWO) on capacited controller to find the optimized position of the controller with minimum link failure.

This research aims to determine the best controller placement that minimises the propagation delay between controller-switch and controller-controller nodes. We focus mainly on reducing the number of controllers required while also improving their placement between controller-switch latency and controller-controller in this effort. In addition, the reliability factor of the controller placement is also concentrated, which is a lack in a lot of literature with minimum latency.

# 3 Problem Formulation

The network topology of SDN represented as graph G where $G(V, E)$. The graph G is an undirected graph with set of vertices V to clone the network nodes and E is the set of edges to clone the links between nodes. Furthermore, the size of the network n = V and $V = S \cup K$ where S is set of switch nodes and K is set of controller nodes.

To overcome the load of the controller the network was partitioned as sub-networks as $Q_i(V_i, E_i)$ where $V_i$ represents the set of nodes of the sub-network and $E_i$ represents the set of links in the sub-network. The formulation representing the relation between network and sub-networks as follows.

$$\bigcup_{i=1}^{c} V_i = V \tag{1}$$

where c is number of controllers.

$$\nexists\, v \in Q_i \cap Q_j \; where \; i \neq j \; and \; v \in V \tag{2}$$

Equation (1) indicates the vertex set of parent network is defined as union of all sub network vertices and Equation (2) provides the disjoint condition of vertex set of sub networks.

## 3.1 Average Delay between Switch Nodes and Controller Nodes

The link weight represented as geodesic distance (shortest path) between switch and controller $d(k, s)$ where $k$ represents the controller node and $s$ represents the switch node.

$$P_{l_{avg}} = \frac{1}{n} \sum_{k_i \in K} \underset{s_j \in S}{Inf}\, d(k_i, s_j) \tag{3}$$

## 3.2 Worst-case Delay between Switch Nodes and Controller Nodes

The worst-case propagation delay or latency of the network between controller node to switch node defined as follows.

$$P_{l_{worst}} = \underset{k_i \in K}{Sup}\; \underset{s_j \in S}{Inf}\, d(k_i, s_j) \tag{4}$$

Where $k$ represents the controller node and $s$ represents the switch node.

### 3.3 Average Delay Between Controller Nodes

The delay between controller to controller nodes also effects the latency of the network. The relation controller nodes represented as follows.

$$P_{l_{avg}}^{K} = \frac{\sum_{i=1}^{c} \min\limits_{k_i, k_j \in K} d(k_i, k_j)}{c(c-1)} \; where \; i \neq j \tag{5}$$

Where $k$ represents the controller node and $s$ represents the switch node.

### 3.4 Network System Reliability

To construct a reliable network based on minimized latency principles, we consider the minimum propagation latency and single link failure. When ever link failure occurs there may be chance of isolating the network nodes from the controller. By considering isolation of network nodes, we formulate a network reliability based on average number of nodes disconnected on single failure.

The average link failure over the entire network is performed

$$\frac{\sum_{i=1}^{m} h(e_i)}{m} \tag{6}$$

where $m$ is number of links in the entire network and $h(e_i)$ is number of nodes which are disconnected when the link $e_i$ fails.

Th reliability over the entire network is calculated using the formulae

$$E_N = m \times R_N \tag{7}$$

where

$$R_N = \frac{\sum_{i=1}^{m} h(e_i) \times \kappa}{m} \tag{8}$$

Here $\kappa$ is the average failure of a single links in 100 time units and we also assumed that every link is having same value of average failure.

By combining equation (7) and (8) we have a much simple formulae for reliability as:

$$E_N = \sum_{i=1}^{m} h(e_i) \times \kappa \tag{9}$$

## 4 Wisdom Of Artificial Crowds GA Algorithm

Wisdom Of Artificial Crowds (WOAC) is a metaheuristic genetic algorithm that provides overall optimization. WOAC is a post-processing technique in which autonomous artificial agents combine their separate solutions to produce a single answer that outperforms all other solutions in the population [19]. The WOAC is a natural phenomena that inspired the algorithm [37].

Genetic algorithms (GA) have a similar basic structure and workflow, but the individual details differ depending on the task. A parent selection method, crossover, and mutation method make up the algorithm.

**Algorithm 1** Basic Genetic Algorithm

```
Input: population with randomly produced chromo-
       somes.
proc Basic_GA()
while terminating condition no reached do
    parents = proc_getParents();
    child = proc_crossover(parents);
    child = proc_mutate(child);
    population.add(child);
end while
```

The network nodes consider generating the initial population required to begin the genetic search. For example, after deciding on the size of the initial population, the necessary number of controller nodes is selected at random. Then, the value of each element in the string representing each alternative solution can be determined using a pseudo-random number generator. Alternatively, pre-computed high fitness strings acquired from previous genetic algorithm runs can be seeded into a population of individuals.

The above Algorithm 1 aims to enhance population fitness by making the population's fittest members generate the most fabulous offspring that deliver a more acceptable solution to the issue. This process persists until a terminating event is met, which could live as simple as running the whole number of generations or as complex as unaltered fitness over a specified number of generations or finding a solution to the issue.

Two distinct parent choosing approaches, a single crossover method, and two diverse mutation processes are all used by the GA. The condition of the population and how near it is to encounter a solution determines which of the parent selection and mutation strategies are chosen. The functions parent selection (Algorithm 2 and Algorithm 3), crossover (Algorithm 4, and mutation (Algorithm 5 and Algorithm 6) are calculated.

The ability to exchange genetic material between superior individuals in a population is crucial for any evolutionary system. In this process, it is required to pass on the beneficial features of the parent solutions to their children. Here, different crossover approaches are considered to have the crucial property of preserving potential partial solutions while retaining sufficient genetic diversity in the solution pool.

Genetic mutations are periodically required to add genetic diversity to highly homogeneous population created by the fittest individuals, allowing for a greater exploration of search space. To acquire any level of genetic diversity, need to repeat the mutation operation in given number of times.

**Algorithm 2** Selecting_parents1

```
1: Input: population
2: Output: parentone , parenttwo
3: proc Selecting_parents1()
4:    tempParents ← randomly chosen chromosomes from
                    the population
5:    parentone ← the fitted parent from tempParents
6:    tempParents ← randomly chosen chromosomes from
                    the population
7:    parenttwo ← the fitted parent from tempParents .
```

**Algorithm 3** Selecting_parents2

```
1: Input: population
2: Output: parentone, parenttwo
3: proc Selecting_parents2()
4:    parentone ← the chromosome with the best perfor-
                  mance
5:    parenttwo ← the chromosome with the best perfor-
                  mance .
```

**Algorithm 4** Crossover

```
1: Input: population
2: Output: child
3: proc Crossover()
4:    crosspoint ← a randomized location on a chromosome
5:    child ← parent 1's location up to and including the
              crosspoint + parent 2's location following
              the crosspoint to the end of the chromosome .
```

**Algorithm 5** mutation1

```
1: Input: population
2: Output: chromosome
3: proc mutation1()
4: for gene in chromosome do
5:    newGene ← randomly selected gene
6:    chromosome ← newGene
7: end for.
```

# 5 Performance Evaluation

The WOACCPP algorithm was tested using the topology of Internet2 OS3E taken from [16]. The cities and high-speed cables are represented by 34 nodes

**Algorithm 6** mutation2

```
Input: population
Output: chromosome
proc mutation2()
for gene in chromosome do
    newGene ← randomly selected gene from selected
              population
    chromosome ← newGene
end for.
```

**Algorithm 7** WisdomOfArtificialCrowds

```
Input: population
proc WisdomOfArtificialCrowds()
 wiseChromosomes ← chromosome with the best per-
                   formance
 keep_gene ← the most desirable half of the eventual
             population
for map in child do
    newgene ← most desirable from keep_gene
    wiseChromosomes ← newgene
end for.
```

**Algorithm 8** Reliability Calculation

```
Input: degree_of_links, average_failure
proc calculation_of_reliability()
for e in edges_of_graph do
  sum ← sum + (degree_of_links*average_failure)
end for
```

$R_c \leftarrow (1/size\_of\_edges) * sum$ .

and 42 linkages in the American network. The propagation latency between two nodes is calculated using the harvesine distance formula considered from [38] and the minimal distances using Dijkstra algorithm shown in [39]. We approximated the results with MDPC proposed by [35] and optimized K- means by [22] to determine the significance of WOACCPP.

The WOACCPP minimizes latency by optimistically placing the controller in the best suitable position such that the network's average latency and worst-case latencies are optimized. This also, minimizes average controller to controller latency with the optimistic shortest path connection between the controllers. In addition, optimal controller placement induces high reliability and minimizes the average switch failures with in the network.

The outstanding performance of the proposed approach, as shown in Table 1, concerning other approaches namely, MDPC and optimized K-means.

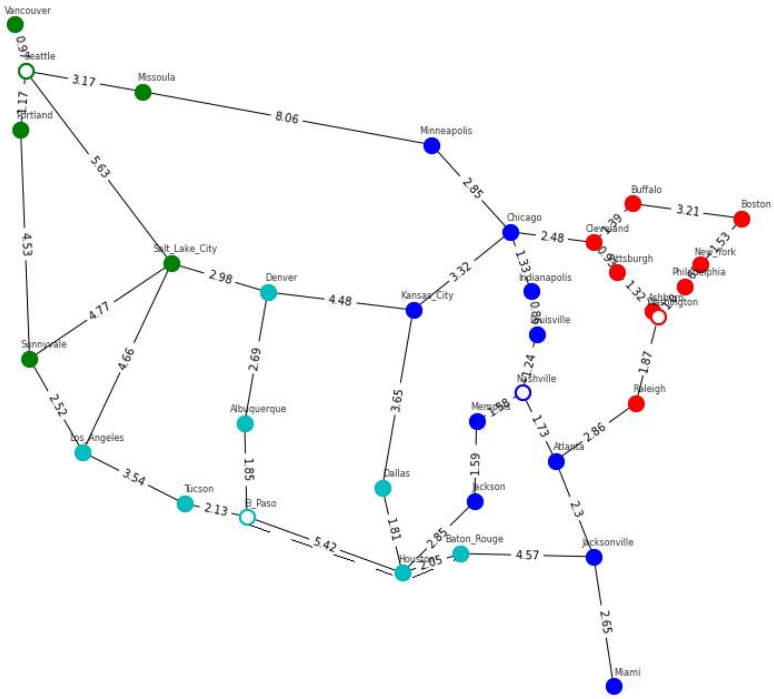


**Fig. 1** network with k = 4 controllers

Figure 1 and Figure 2 represent best placement of controllers with OS3E network topology on k=4 controllers and k=5 controllers. From Figure 1 and Figure 2, empty circles are represented as centers, filled circles are switch nodes, and the link between switches and centers is represented as latency. The double lines reflect the network's worst-case latency.

The WOACCPP's average network latency is compared to the MDPC and the optimized K-means algorithms to understand the performance of the proposed approach. For example, with k = 4 centres, the average latency of switch device to controller is 3 ms for the proposed approach, 3.2 ms for MDPC, and 3.5 ms for optimized K-means, as shown in Figure 3. The WOACCPP outperforms MDPCs, optimized K-means even at k = 5 also. In WOACCPP, the latency is reduced by 10.71 per cent compared to existing solutions, namely MDPC, optimized K-means.

The Figure 4 depicts the worst-case latency of switch node to controller device. WOACCPP's worst-case latency at k=7 outperforms MDPC and optimised K-means performance. The WOACCPP took 4.5% faster than both MDPC and optimised K-means.

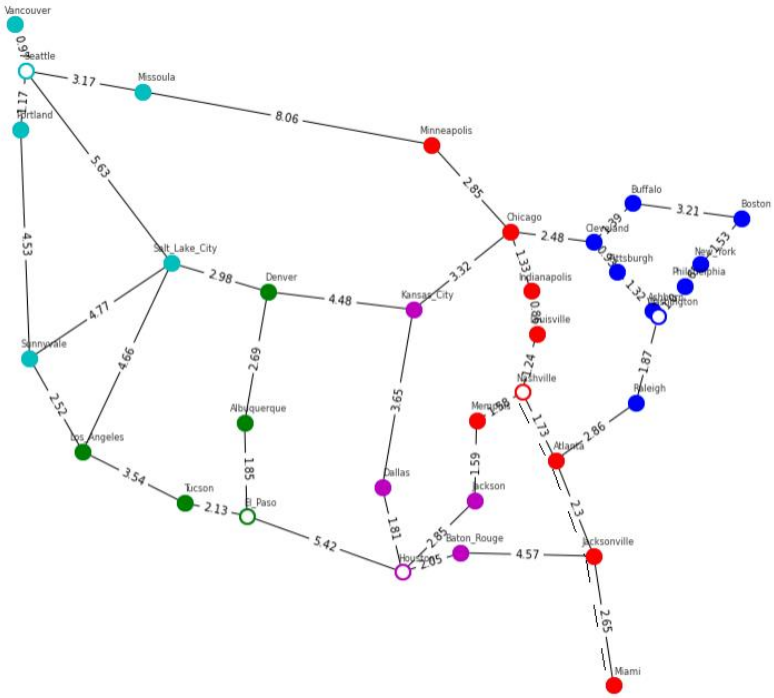


**Fig. 2** network with k = 5 controllers

The latency between the controller nodes is also crucial in the CPP problem. The average latency between controllers is depicted in Figure 5. When k = 7 controllers, the WOACCPP gives a decent average latency between them. WOACCPP has a minimum latency of 4.2 milliseconds, smaller than the 4.4 milliseconds of each MDPC and optimised K-means.

The network reliability $E_N$ is compared with increasing the failure probabilities $\kappa$. The Figure 6 shows the reliability, WOACCPP is more reliable than other MDPC and optimized K- means. Here, minimum the $E_N$ value higher the reliability.

The average network switch failures of the network showed in Figure 7 by increasing the failure probabilities for every 100 units of time. The average switch failures are minimum in WOACCPP.

Figure 8 shows the reliability of the network by raising the number of controllers by assuming the $\kappa$ value is 1 for every 100 units of time. The $E_N$ value is lowered by growing the controller. The proposed approach gives a minimum $E_N$ value compared to MDPC and optimized K-means. The minimum the $E_N$ value is more the reliability.

**Table 1** Performance metrics

| | S to C[1] | | C to C[2] | | Average |
|---|---|---|---|---|---|
| Method | Average Latency | Worst-case Latency | Average Latency | Reliability | Switch Failure |
| WOACCPP | 3.0 (k[3]=4) | 4.5 (k=7) | 11.7 (k=6) | 57 (k=4) | 1.35 (k=4) |
| MDPC | 3.2 (k=4) | 5 (k=7) | 12 (k=6) | 59 (k=4) | 1.4 (k=4) |
| Optimized K-means | 3.5 (k=4) | 5 (k=7) | 12.1 (k=6) | 62 (k=4) | 1.47 (k=4) |

[1]Switch to Controller

[2]Controller to Controller

[3]k is number of controllers

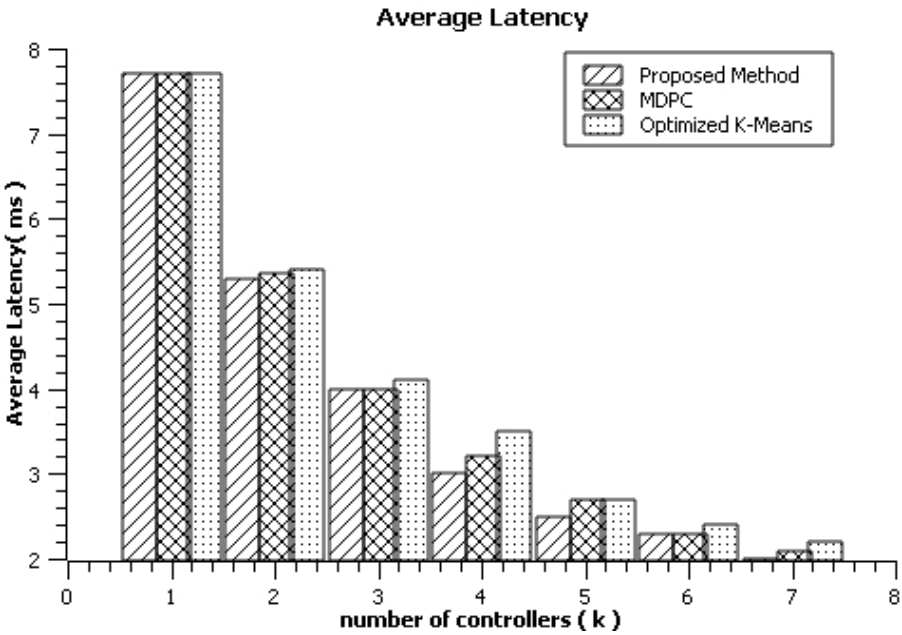


**Fig. 3** average latency of network

Figure 9 illustrates the average switch failures by raising the controllers and preserving the $\kappa$ value 1 for every 100 units of time. The average switch failures of the proposed approach live minimum over MDPC and optimized K-means.

# 6 Conclusion

The main problem faced by software defined networks is placement of controller, which deals latency and reliability problems. To place the Controller in optimized location, authors are using Wisdom of Artificial Crowds (WOAC)

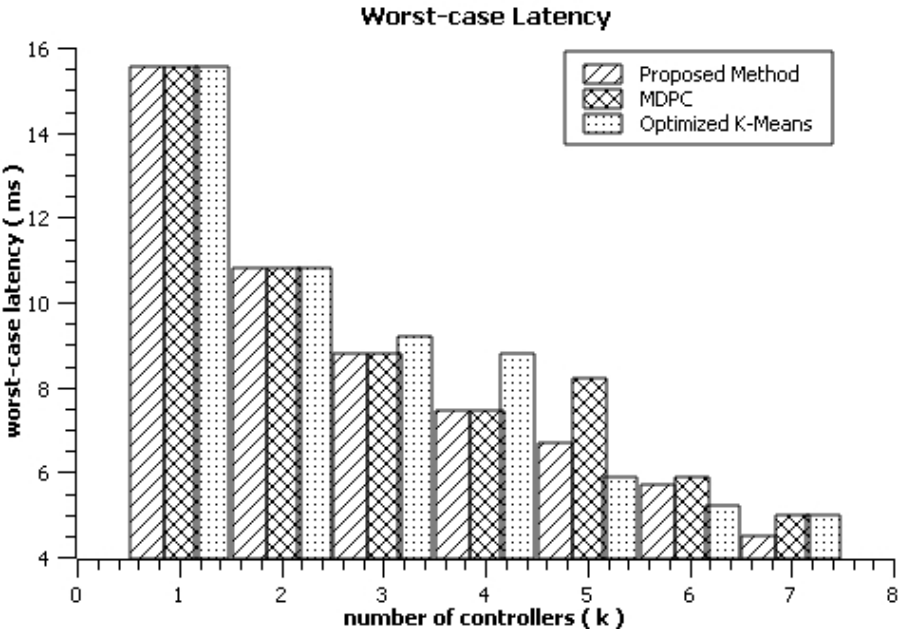


**Fig. 4** worst-case latency of network

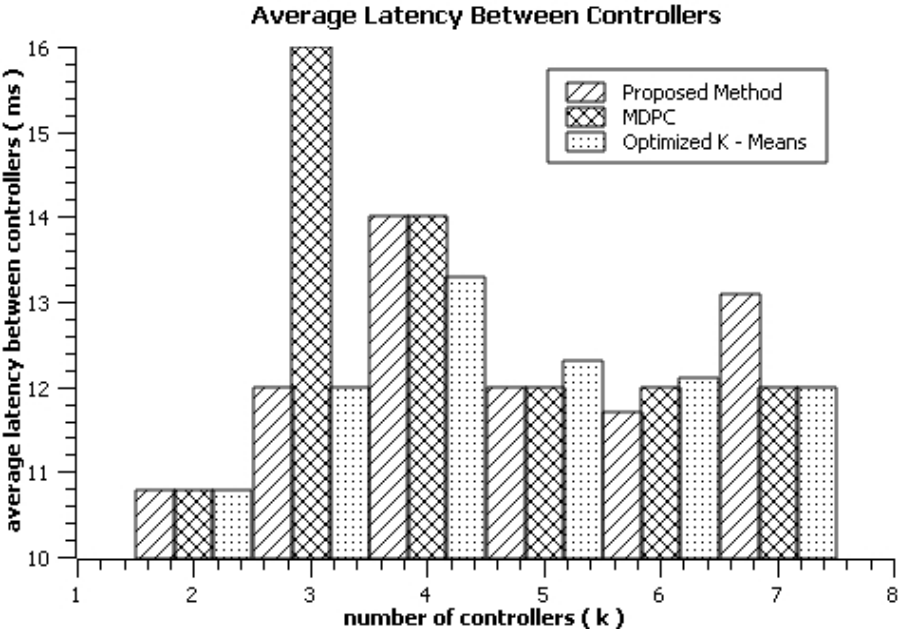


**Fig. 5** average latency between controllers of the network

based genetic algorithm is used and proposed a WOACCPP (Wisdom Of Artificial Crowds Controller Placement Problem). The proposed approach tested

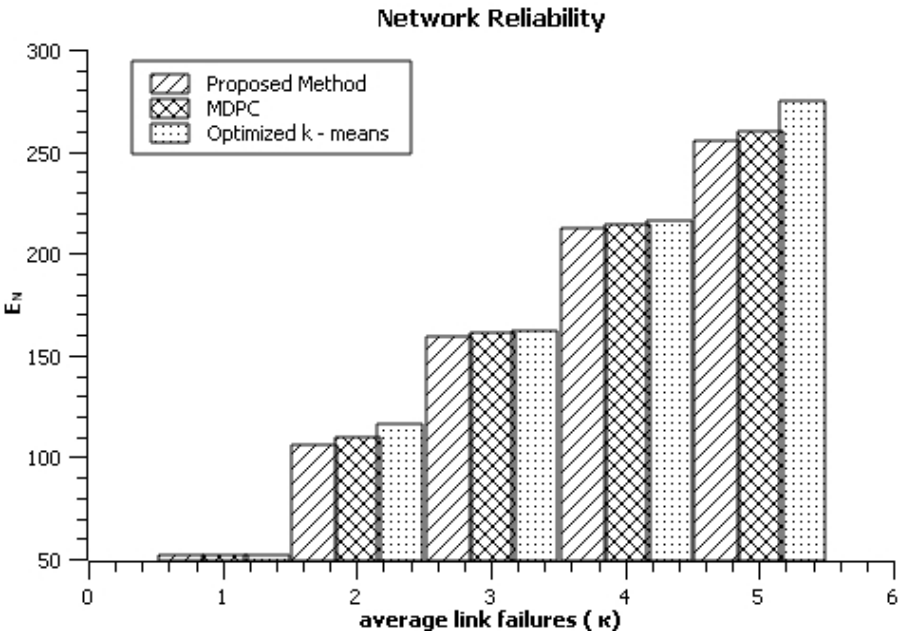


**Fig. 6** over all network reliability

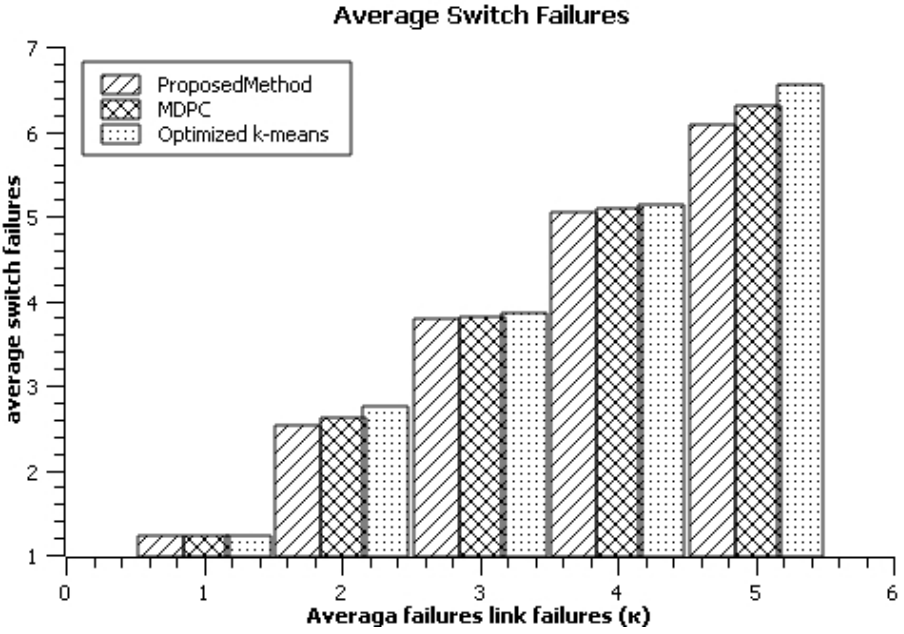


**Fig. 7** average switch failures

with Multiple controllers to check the better scalability and reliability in

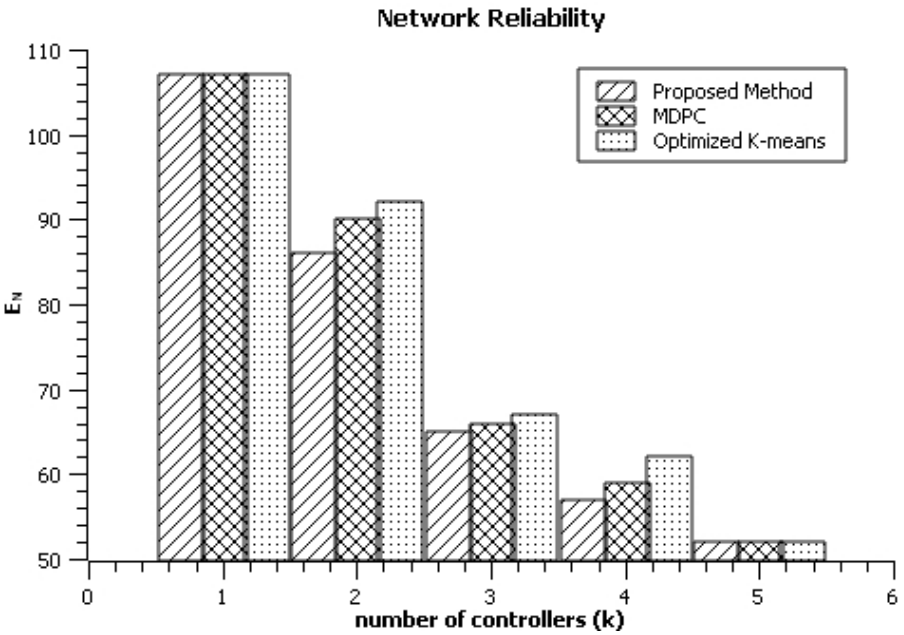


**Fig. 8** network reliability on controllers

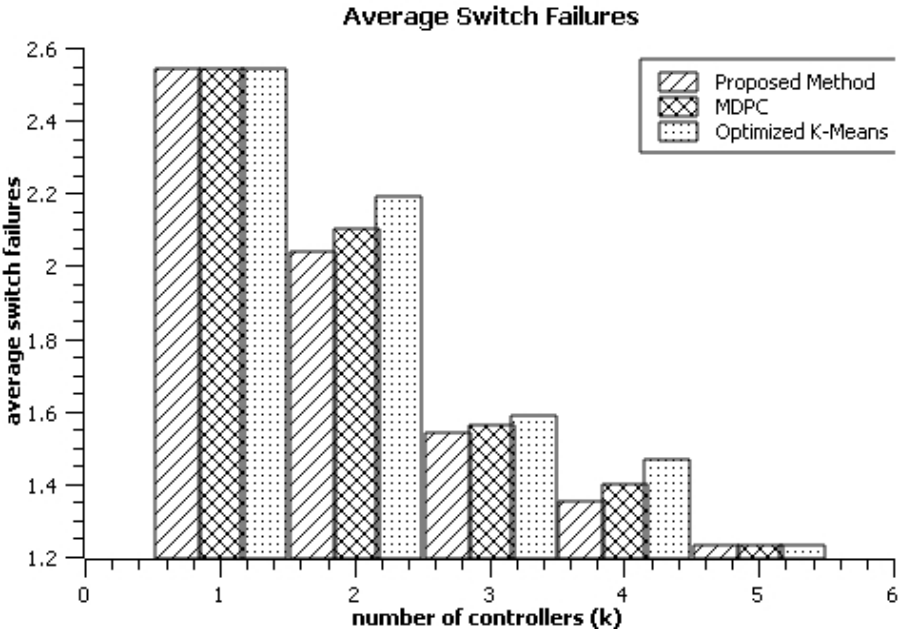


**Fig. 9** average switch failure on Controllers

SDN. Internet2 OS3E topology is considered to test and check the performance of proposed approach. The proposed controller placement technique

reduces latency and improves reliability by comparing the existing solutions namely Modified Density Peak Clustering (MDPC) and optimized K-means approaches. In future, we planning to concentrate other parameters such as availability of controller to improve the performance of the network.

# 7 Declarations

## 7.1 Ethical Approval and Consent to participate

Not applicable.

## 7.2 Human and Animal Ethics

Not applicable.

## 7.3 Consent for publication

Not applicable.

## 7.4 Availability of supporting data

Used publicly available data-set http://www.topology-zoo.org/dataset.html

## 7.5 Competing interests

The authors declare that they have no competing interests.

## 7.6 Funding

Not applicable.

## 7.7 Authors' contributions

Vidya Sagar Thalapala and Anandhu Mohan are contributed mathematical formulation and implementation details for proposed work. Vidya Sagar Thalapala and Dr. Koppala Guravaiah wrote & reviewed the manuscript.

## 7.8 Acknowledgements

I would like express special thanks to IIIT Kottayam for providing the lab facilities to do the experimentation.

## 7.9 Authors' information

Vidya Sagar Thalapala: He obtained his degree in Computer Science and Engineering in 2002 from Narayana Engineering College, J.N.T.U. Hyderabad and Post graduate degree in Computer Science and Engineering in 2011 from Acharya Nagarjuna University, Andhra Pradesh, India. He is currently doing research in Indian Institute Information Technology, Kottayam, Kerala, India.

His research interest includes Software Defined Networking and Routing in Mobile Ad Hoc Networks.

Anandhu Mohan: Obtained bachelor of science degree as well as the master of science degree in Mathematics from Mahatma Gandhi University Kottayam, India. Currently pursuing PHD in Mathematics from department of computational science and humanities, Indian institute of information technology Kottayam, India. The interested areas of research are fuzzy graph theory and coding theory.

Dr. Koppala Guravaiah, working as Assistant Professor at Indian Institute of Information Technology (IIIT-K), Pala, Kerala, India. Completed Ph.D. on the topic performance of Routing protocols in Wireless Sensor Networks using River formation dynamics from National Institute of Technology Tiruchirappalli, India. His research interests include the Internet of Things (IoT), Software Defined Networks, Wireless Ad hoc, and Sensor network routing protocols and their applications. He published 5 Journals, 8 conferences, 2 book chapters in the areas mentioned above by his name.

## References

[1] Clemm, A., Zhani, M.F., Boutaba, R.: Network Management 2030: Operations and Control of Network 2030 Services. Journal of Network and Systems Management **28**(4), 721–750 (2020). https://doi.org/10.1007/s10922-020-09517-0. Accessed 2022-04-01

[2] Selvi, K., Thamilselvan, R.: An intelligent traffic prediction framework for 5g network using sdn and fusion learning. Peer-to-Peer Networking and Applications, 1–17 (2022)

[3] Farhady, H., Lee, H., Nakao, A.: Software-defined networking: A survey. Computer Networks **81**, 79–95 (2015)

[4] Karakus, M., Durresi, A.: Quality of Service (QoS) in Software Defined Networking (SDN): A survey. Journal of Network and Computer Applications **80**, 200–218 (2017). https://doi.org/10.1016/j.jnca.2016.12.019. Accessed 2022-04-23

[5] Xu, X., Wang, S., Li, Y.: Identification and predication of network attack patterns in software-defined networking. Peer-to-Peer Networking and Applications **12**(2), 337–347 (2019). https://doi.org/10.1007/s12083-017-0629-6. Accessed 2022-05-13

[6] Masoudi, R., Ghaffari, A.: Software defined networks: A survey. Journal of Network and Computer Applications **67**, 1–25 (2016). https://doi.org/10.1016/j.jnca.2016.03.016. Accessed 2022-04-23

[7] Wang, G., Zhao, Y., Huang, J., Wang, W.: The controller placement problem in software defined networking: A survey. IEEE Network **31**(5), 21–27 (2017)

[8] Blial, O., Ben Mamoun, M., Benaini, R.: An overview on sdn architectures with multiple controllers. Journal of Computer Networks and Communications **2016** (2016)

[9] Hu, T., Guo, Z., Yi, P., Baker, T., Lan, J.: Multi-controller based software-defined networking: A survey. IEEE access **6**, 15980–15996 (2018)

[10] Singh, S., Jha, R.K.: A survey on software defined networking: Architecture for next generation network. Journal of Network and Systems Management **25**, 321–374 (2016)

[11] Moufakir, T., Zhani, M.F., Gherbi, A., Bouachir, O.: Collaborative Multi-domain Routing in SDN Environments. Journal of Network and Systems Management **30**(1), 23 (2022). https://doi.org/10.1007/s10922-021-09638-0. Accessed 2022-04-01

[12] Das, T., Sridharan, V., Gurusamy, M.: A survey on controller placement in sdn. IEEE Communications Surveys & Tutorials **22**(1), 472–503 (2019)

[13] Zhang, Y., Cui, L., Wang, W., Zhang, Y.: A survey on software defined networking with multiple controllers. Journal of Network and Computer Applications **103**, 101–118 (2018). https://doi.org/10.1016/j.jnca.2017.11.015. Accessed 2022-04-23

[14] Sahay, R., Meng, W., Jensen, C.D.: The application of Software Defined Networking on securing computer networks: A survey. Journal of Network and Computer Applications **131**, 89–108 (2019). https://doi.org/10.1016/j.jnca.2019.01.019. Accessed 2022-04-23

[15] Heller, B., Sherwood, R., McKeown, N.: The controller placement problem. ACM SIGCOMM Computer Communication Review **42**(4), 473–478 (2012)

[16] Internet2 open science, scholarship and services exchange. https://internet2.edu/network/ Accessed 2022-03-11

[17] Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103 (1972). Springer

[18] Rabanal, P., Rodríguez, I., Rubio, F.: Using river formation dynamics to design heuristic algorithms. In: International Conference on Unconventional Computation, pp. 163–177 (2007). Springer

[19] Yampolskiy, R.V., El-Barkouky, A.: Wisdom of artificial crowds algorithm for solving np-hard problems. International Journal of Bio-inspired computation **3**(6), 358–369 (2011)

[20] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. ADDISON-WESLEY PUBLISHING COMPANY, INC, Boston, MA (1989)

[21] Yao, G., Bi, J., Li, Y., Guo, L.: On the capacitated controller placement problem in software defined networks. IEEE communications letters **18**(8), 1339–1342 (2014)

[22] Wang, G., Zhao, Y., Huang, J., Duan, Q., Li, J.: 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 (2016). IEEE

[23] Zhong, Q., Wang, Y., Li, W., Qiu, X.: A min-cover based controller placement approach to build reliable control network in sdn. In: NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, pp. 481–487 (2016). IEEE

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

[25] Killi, B.P.R., Rao, S.V.: Capacitated next controller placement in software defined networks. IEEE Transactions on Network and Service Management **14**(3), 514–527 (2017)

[26] Sanner, J.-M., Hadjadj-Aoul, Y., Ouzzif, M., Rubino, G.: An evolutionary controllers' placement algorithm for reliable sdn networks. In: 2017 13th International Conference on Network and Service Management (CNSM), pp. 1–6 (2017). IEEE

[27] Sahoo, K.S., Puthal, D., Obaidat, M.S., Sarkar, A., Mishra, S.K., Sahoo, B.: On the placement of controllers in software-defined-wan using meta-heuristic approach. Journal of Systems and Software **145**, 180–194 (2018)

[28] Ishigaki, G., Gour, R., Yousefpour, A., Shinomiya, N., Jue, J.P.: Cluster leader election problem for distributed controller placement in sdn. In: GLOBECOM 2017-2017 IEEE Global Communications Conference, pp. 1–6 (2017). IEEE

[29] Wang, G., Zhao, Y., Huang, J., Wu, Y.: An effective approach to controller placement in software defined wide area networks. IEEE Transactions on Network and Service Management **15**(1), 344–355 (2017)

[30] Chai, R., Yuan, Q., Zhu, L., Chen, Q.: Control plane delay minimization-based capacitated controller placement algorithm for sdn. EURASIP Journal on Wireless Communications and Networking **2019**(1), 1–17 (2019)

[31] Torkamani-Azar, S., Jahanshahi, M.: A new gso based method for sdn controller placement. Computer Communications **163**, 91–108 (2020)

[32] Abdi, A., Hosseini Seno, S.A., et al.: Controller placement in software defined network using iterated local search. Journal of Artificial Intelligence and Data Mining **8** (2020)

[33] Dhar, M., Bhattacharyya, B.K., Kanti Debbarma, M., Debbarma, S.: A new optimization technique to solve the latency aware controller placement problem in software defined networks. Transactions on Emerging Telecommunications Technologies **32**(10), 4316 (2021)

[34] Rasol, K.A.R., Domingo-Pascual, J.: Joint latency and reliability-aware controller placement. In: 2021 International Conference on Information Networking (ICOIN), pp. 197–202 (2021). IEEE

[35] Qi, Y., Wang, D., Yao, W., Li, H., Cao, Y.: Towards multi-controller placement for sdn based on density peaks clustering. In: ICC 2019-2019 IEEE International Conference on Communications (ICC), pp. 1–6 (2019). IEEE

[36] Kanodia, K., Mohanty, S., Kurroliya, K., Sahoo, B.: Ccpgwo: A meta-heuristic strategy for link failure aware placement of controller in sdn. In: 2020 International Conference on Inventive Computation Technologies (ICICT), pp. 859–863 (2020). IEEE

[37] Surowiecki, J.: THE WISDOM OF CROWDS. DOUBLEDAY, New York, NY (2004)

[38] Robusto, C.C.: The cosine-haversine formula. The American Mathematical Monthly **64**(1), 38–40 (1957)

[39] Dijkstra, E.W., *et al.*: A note on two problems in connexion with graphs. Numerische mathematik **1**(1), 269–271 (1959)