Metaheuristic Techniques for Controller Placement in Software-Defined Networks

Sagarika Mohanty
Dept. of CSE
NIT Rourkela
Odisha, India
sagarikam_23@yahoo.com

Prateekshya Priyadarshini
Dept. of CSEA
IGIT Sarang
Dhenkanal, Odisha, India
prateekshyapriyadarshini@gmail.com

Sampa Sahoo
Dept. of CSE
NIT Rourkela
Odisha, India
sampaa2004@gmail.com

Bibhudatta Sahoo
Dept. of CSE
NIT Rourkela
Odisha, India
bdsahu@nitrkl.ac.in

Srinivas Sethi
Dept. of CSEA
IGIT Sarang
Dhenkanal, Odisha, India
srinivas_sethi@igitsarang.ac.in

***Abstract*—Software defined networks provides a global network view with centralized management. To maintain the network configuration, multiple controllers are required. The network performance depends on the optimal number of controllers and their placement.Due to the large size and complexity involved, meta-heuristic algorithms are the probable choice that can solve the problems in an acceptable amount of time. This paper addresses the controller placement problem in SDN by using two meta-heuristic techniques. The objective is to find optimal number and location of controllers in the network while minimizing the propagation latency and optimizing cost. A random approach is adopted for initial placement of controllers. The assignment of switches to the controllers is done based on their shortest distance. Then an efficient genetic algorithm based placement solution is proposed to find the optimal location of controllers which minimizes cost. Our proposed genetic algorithm is different from the standard genetic algorithm in terms of generation and replacement for determining the best cost and the optimal location of controllers . The same experiment is done on simulated annealing (SA) and random method. For evaluation purpose, we have used some real topologies. The results of our enhanced GA performs better compared to simulated annealing and random placement approach.**

***Index Terms*—Software Defined Network, Controller Placement Problem, Genetic Algorithm, Simulated Annealing , Latency**

I. INTRODUCTION

Software Defined Network (SDN) is distinguished from the traditional IP network with the centralized control. This enables network service providers and operators a chance to make innovation, programmability and flexibility for better network design. In SDN technology, the control plane is responsible for management of network nodes, monitoring and updates the table information of switches through flow forwarding. The network intelligence is separated from the hardware and given to one or more external intelligent elements called controllers [1] [2]. To maintain the performance and scalability for a large network definitely multiple controllers are required. In a multi-controller scenario, network performance depends on the number and location of controllers. The placement of controllers p in a network of n nodes where $p << n$, this problem is definitely a hard problem and a variation of the facility or warehouse location problem. For example, in Janet Backbone network with 29 nodes, placement of 3 controllers require 3654 combinations. Due to large size, memory requirement and complexity involved meta-heuristic techniques are usually the choice of researchers which explore a subset of the total search space by giving a feasible solution (either optimal or sub optimal) within an acceptable period of time.

This paper addresses the controller placement problem in SDN by using some meta-heuristic techniques. Considering the complexity of wide area SDN, an efficient genetic algorithm is proposed to find an effective solution. Given n switches with their locations, the controller placement problem will find out a solution of choosing p controllers and their locations. This will minimize the propagation latency between the switches and the controllers while reducing the total cost of the network. The contributions and the sequence of the proposed work are elaborated as follows:

- Random placement of controllers
- Allocation of switches to the controllers based on their shortest distance.
- Use an enhanced genetic algorithm for controller placement and compare its result with simulated annealing and random placement algorithm.
- The objective is to determine a placement solution which minimizes the latency while reducing the cost of the network.
- Consider some real datasets from the topology zoo and apply the techniques on these topologies.

The next section presents the related work. Problem statement and the proposed enhanced genetic algorithm is discussed in

978-1-7281-1895-6/19/$31.00

section III.

Section IV analyzes the performance of the proposed GA algorithm on some real topologies. This section also presents the comparison of this algorithm with simulated annealing and randomized algorithm. Section V concludes the paper.

II. RELATED WORK

Brandon Heller et al. [3] considered latency for the placement problem. M.F.Bari et al. [4] formulated a framework to connect switches to the controllers which reduces flow set up time and communication overhead. G. Yao et al. [5], emphasized on minimizing the maximum latency. According to them for an extensive network, multiple domains with multiple controllers are necessary. Sallahi et al. [6] proposed a model to find the optimal number of controllers, their locations while considering various controller types. Lange et al. [7] applied heuristics to place k controllers in a large network. Ros et al. [8] finds the placement with minimum cost by applying heuristics. In POCO framework Hock et al. [9] performed an evaluation considering different metrics. Like Hu et al. [10] they consider the resiliency aspect. Killi et al. [11] proposed mathematical formulation considering two indexed and three indexed variable in case of controller failure. They used heuristics to solve the problem. In MOCO framework Jalili et al. [12] considered a number of performance metrics. According to them a trade off is required between these competing metrics. A density based clustering approach is proposed by Liao et al. [13] to find the number of controllers.Hu Bo et al. [14] considered the load diversity factor in a distributed environment and used multi objective genetic algorithm to get the connection between network elements and applied minimum delay algorithm for placement. The authors in [15] proposed a multiobjective genetic algorithm considering different performance metrics. V Huang et al. [16] proposed a new algorithm that integrates genetic algorithm and gradient descent to achieve highest control plane utilization and low response time.

III. PROBLEM STATEMENT

We represent the network as a graph $G(V, E)$ in which the node set $V = S \cup C$, where $S = \{x_1, x_2, ..., x_n\}$ be the switch set, $C = \{y_1, y_2, ..., y_p\}$ be the controller set and p be the potential controller sites, $E = \{e_1, e_2, ...\}$ be the set of communication links connecting these nodes, f_y be the fixed cost of placing a controller at the desired location, l_x be the load on the controller i.e., the packets per second forwarded from a switch to a controller, K_y be the processing capacity of a controller, d_{xy} be the shortest distance from switch x to controller y.

In a network with p controllers, the rest nodes (switches) will be allocated to their nearest controller considering the load capacity. Hence some cost is associated with it. In this, the objective is to explore the optimal location of controllers in the network so that latency will be reduced between the switches and controllers which will lead to the reduce cost of the system. The objective function is represented as:

$$\min \sum_{y \in C} f_y Z_y + \sum_{x \in S} \sum_{y \in C} l_x d_{xy} W_{xy} \tag{1}$$

This is subject to

$$\sum_{y \in C} Z_y = p \tag{2}$$

$$\sum_{y \in C} W_{xy} = 1 \quad \forall x \in S \tag{3}$$

$$W_{xy} \leq Z_y \quad \forall x \in S, \quad \forall y \in C \tag{4}$$

$$\sum_{x \in S} l_x W_{xy} \leq K_y Z_y \quad \forall y \in C \tag{5}$$

$$W_{xy} \in \{0, 1\}, \quad \forall x \in S, \quad \forall y \in C \tag{6}$$

$$Z_y \in \{0, 1\} \quad \forall y \in C \tag{7}$$

Equation (1) computes the setup cost for placing the controllers at the probable location and the routing cost of connecting the switches with their controllers. The objective is to minimize it. Equation (2) guarantees that p controllers are placed in the network. Equation (3) ensures that each switch is allocated to a controller. Equation (4) restricts a switch from assignment to a position if the controller is not deployed at that particular location. The load of the switches incurred on a controller should be within the controller's capacity as described by Equation (5). Equation (6) and (7) are decision variables. $W_{xy} = 1$, if switch x is allocated to controller y and $Z_y = 1$, that is a controller is located at position y , otherwise 0.

IV. PROPOSED METAHEURISTIC TECHNIQUES

The genes of a chromosome represent a solution of controller placement. For example, for $p = \{0, 4, 7, 11\}$ is a chromosome where $0, 4, 7, 11$ are the node id or the possible controller locations of a particular topology. For example, for Janet Backbone network with $n = 29$, $p = 3$ and population size $npop = 15$, each member of the population should include node indices from $\{0, 1, 2, ..., 28\}$. Each member represents a placement solution such as $\{3, 11, 23\}, \{6, 19, 26\}, ...$ for 15 number of times.

A. Proposed Genetic Algorithm (GA)

The shortest path matrix is calculated from the latitude and longitude information of the network. The population size, $npop$ is the cardinality of the population, pop. For each member m_i of the population randomly select p controllers from n nodes and assign switches to the controller using their shortest distance. Then the objective function for each member, $m\_cost(i)$ is calculated $npop$ times using equation (1). The best cost b_cost and the worst cost w_cost is computed from the population.

Our proposed genetic algorithm is quite different from the standard genetic algorithm. In selection, two members m_1, m_2 are selected randomly from the initial population. This process is different from the general genetic algorithm where roulette

Algorithm 1 Calculation of Objective function

Input: srt_path_mat
Output: m_cost, b_cost, w_cost
1: $p \leftarrow$ Number of Controllers
2: $pop \leftarrow$ Population
3: $npop \leftarrow$ Population size and $npop = |pop|$
4: $m_i \leftarrow i^{th}$ member in pop where $m_i \leftarrow pop(i), i = 1, 2, ..., npop$
5: **for** $i = 1$ to $npop$ **do**
6: $m\_cost(i) \leftarrow \emptyset$
7: randomly select p controllers from n
8: **for** $j = 1$ to p **do**
9: assign every switch to its nearest controller using srt_path_mat
10: **end for**
11: calculate $m\_cost(i)$ using equation (1)
12: **end for**
13: $b\_cost \leftarrow \min(m\_cost)$
14: $w\_cost \leftarrow \max(m\_cost)$

wheel is used for selection. For our experiments, we have considered both roulette wheel and random method. As random method is giving better results so, we have used random approach for selection. Then run the generation operator [17] on these members to get a new member which is elaborated in algorithm 3. The generation operator is different from the crossover of the standard genetic algorithm. Let s_m be the same node id or controller locations of members m_1, m_2 and d_m be the different node id of m_1, m_2.

Algorithm 2 Proposed Genetic Algorithm

Input: $pop, b\_cost, w\_cost, m\_cost, p$
Output: bf_cost, opt_loc
1: $m\_itr \leftarrow$ maximum number of iterations
2: **for** $i = 1$ to m_itr **do**
3: $m_1, m_2 \leftarrow selection(pop)$
4: $m_{new} \leftarrow generation(m_1, m_2, p)$
5: $m'_{new} \leftarrow mutation(m_{new})$
6: $[pop, m'_{new}\_cost] \leftarrow replacement(m'_{new}, w\_cost, m\_cost)$
7: $[b\_cost, m_b] \leftarrow \min(m\_cost)$
8: $w\_cost \leftarrow \max(m\_cost)$
9: $i = i + 1$
10: **end for**
11: $bf\_cost \leftarrow b\_cost$
12: $opt\_loc \leftarrow m_b$

A draft member m_{dft} is created by taking the union of s_m and d_m. For each node of d_m calculate the node degree, sort this in increasing order and store it in nds. Then from m_{dft} delete the node with lesser node degree one by one that belongs to d_m until the length of m_{dft} equals p (lines 5-15). In this process, a new member m_{new} is generated.
Mutation is done on the genes of the new member m_{new} with certain mutation probability. The new member after mutation m'_{new} will go through the replacement operation.

First compute the objective function of the new member m'_{new} using equation (1). If m'_{new}_cost is higher than w_cost then discard the member. If m'_{new}_cost is equal to any of the member's m_cost then also discard this. Otherwise update the population pop by adding the new member m'_{new}. The minimum of m_cost is the b_cost and its member m_b. The maximum of m_cost is the w_cost. Algorithm 2 iterate for m_itr number of times. The final solution is the best cost bf_cost and the member with its optimal location of controllers opt_loc.

Algorithm 3 generation(m_1, m_2, p)

Input: m_1, m_2, p
Output: m_{new}
1: $s_m \leftarrow$ same index of m_1, m_2
2: $d_m \leftarrow$ different index of m_1, m_2
3: $m_{dft} \leftarrow s_m \cup d_m$
4: $len = length(m_{dft})$
5: **for** $each\ node\ no \in d_m$ **do**
6: $nds = degree(no)$
7: **end for**
8: $nds = sort(nds)$
9: **while** $len > p$ **do**
10: **for** $each\ node\ no \in m_{dft}$ **do**
11: $m_{dft} \leftarrow m_{dft} - nds[1]$
12: $nds \leftarrow nds - nds[1]$
13: **end for**
14: $len = len - 1$
15: **end while**
16: $m_{new} \leftarrow m_{dft}$

Algorithm 4 replacement $(m'_{new}, w\_cost, m\_cost)$

Input: m'_{new}, m_cost, w_cost
Output: pop, m'_{new}_cost
1: $m'_{new}\_cost \leftarrow$ calculate $m\_cost(m'_{new})$ using equation (1)
2: **if** $w\_cost \leq m'_{new}\_cost$ **then**
3: discard m'_{new}_cost
4: **else if** $m'_{new}\_cost == m\_cost(i)$ **then**
5: discard m'_{new}_cost
6: **else** $pop \leftarrow pop \cup m'_{new}$
7: **end if**

B. Controller Placement using Simulated Annealing

Simulated Annealing is an optimization meta-heuristic inspired by the cooling of solids with a low energy configuration. The new solution is accepted depending on the difference in objective function and temperature [18].

- A random initial solution of controller placement is created and assignment of switches to their nearest controller is done as in algorithm 1. The objective (cost) is calculated as in equation (1).

- Then a new solution is generated either by relocating the switch to a different controller or by interchanging two switches connected to different controllers. Then calculate its objective function using equation (1).
- The new placement solution is accepted if its cost is lower compared to the previous one. Otherwise it is accepted if $r < exp^{(-\Delta C/t)}$ where r is a random number $[0, 1]$. ΔC is the difference between the neighboring cost function and t is the control parameter. The algorithm starts with a high temperature and gradually lowered till the stopping condition or final temperature is met using a cooling schedule.

V. RESULT AND ANALYSIS

For implementation, the program is written in MATLAB 2015 and run it on a machine loaded with Intel(R) Core(TM) i7 processors with 64 GB RAM. For our experiment we have taken the data from the topology zoo [19]. It is a store-house of various datasets available in the graphical format. The graphML carry information about the nodes. Haversine formula gives the shortest distance between two points on the surface of the sphere. Haversine formula:

$$hv(\frac{d}{ra}) = hv(\theta_2 - \theta_1)\cos(\theta_1)\cos(\theta_2)hv(\mu_2 - \mu_1) \quad (8)$$

and

$$d = 2ra\ \sin^{-1}(\sqrt{h})$$

where

$$h = hv(\frac{d}{ra})$$

Thus,

$$d = 2ra\sin^{-1}\left(\sqrt{\sin^2(\tfrac{\theta_2-\theta_1}{2}) + \cos(\theta_1)\cos(\theta_2)\sin^2(\tfrac{\mu_2-\mu_1}{2})}\right) \quad (9)$$

make sure that $h \in [0, 1]$
where hv is the haversine function.
$d \leftarrow$ distance between two nodes,
$ra \leftarrow$ radius of the sphere,
$\theta_1, \theta_2 \leftarrow$ latitude of two nodes and
$\mu_1, \mu_2 \leftarrow$ longitude of two nodes.

We have considered three networks Janet Backbone, Sanet and IRIS for our analysis. After running the program a number of times we have set the parameters for our final evaluation. The population size is fixed to 20. The initial and final temperatures are taken as 10 and 0.001. The cooling ratio is set as 0.95 and number of iterations considered as 500 for the evaluation. We analyze our work considering three aspects. In the first aspect the performance of objective function is related to the number of controllers. The impact of the execution time on the number of controllers is taken as the second aspect and the last is to find the optimal location of controllers.

A. Performance based on Objective Function

Fig. 1 show the results of objective function (cost) on the number of controllers. The cost includes the fixed cost for placing the controllers and the routing cost of connecting the switches with the controllers. The fixed cost will increase along with the rise in the number of controllers. But propagation latency will decease with the increasing number of controllers. For GA, in Janet backbone network cost is increasing after 6 controllers. While for Sanet and IRIS networks for GA, cost is increasing after 7 controllers. From the figure it is clearly shown that GA performs better and gives very good results compared to SA and random.

B. Performance based on Execution time

Fig. 2 illustrates the impact of execution time on the number of controllers. As can be seen from figure our enhanced proposed GA converges faster compared to SA. Random algorithm behaves totally different.

C. Optimal location of Controller

Fig. 3 and 4 shows the optimal (sub-optimal or feasible) location of controllers within the network topology. Here the blue dots indicate controller locations and the yellow dots show the rest of the nodes or switches. Considering the objective function that the network will minimize the propagation latency while optimizing cost, Fig. 3 and 4 shows the location of controllers for GA and SA respectively. From experimental analysis, it can be shown that for Janet backbone network 3 to 4 controllers are sufficient while for Sanet and IRIS networks 4 to 5 are sufficient to maintain the network configuration. The maximum limit is within 7. We are not considering the controller positions where either no switch is connected or very few are connected. In this case we have done the reassignment of switches to other nearest controllers considering the controller's capacity.

D. Comparative study

The proposed GA is compared with the GAPA algorithm of Yuan et al. [20] to find the average delay for GERMANY50 network. The algorithm presented by Yuan et al. is modified to suit our working environment. Our proposed GA is giving better result as shown in Fig. 5.

VI. CONCLUSION

While designing a network architecture such as SDN, it is a great challenge for the administrators to determine the desired number of controllers and their location in the network. Networks with a finite number of switches and their locations, the controller placement problem determine the number of controllers and their locations to maximize performance. The objective is to place the controllers in their optimal locations so that the propagation latency between switches and controllers is minimized which will lead to the decrease cost of the network. For initial placement of controllers we have considered random approach and assigned switches based in their shortest distance. The algorithm has been tested and verified by taking

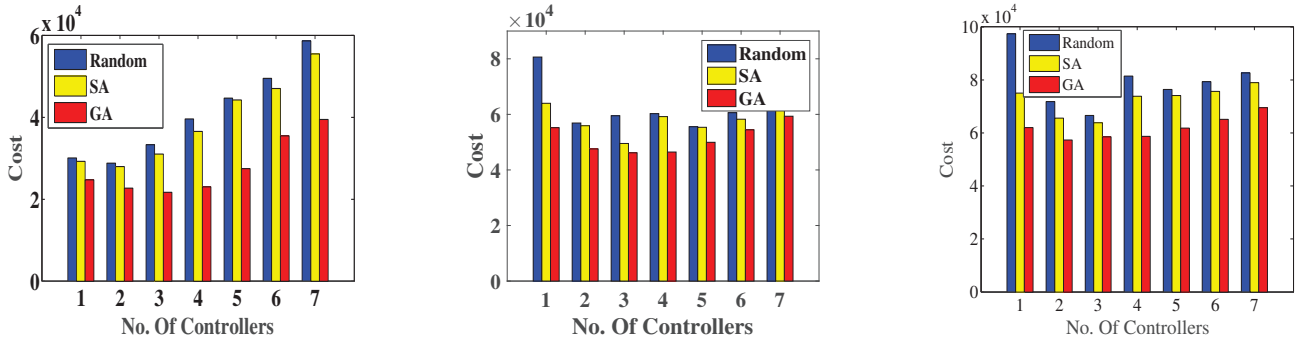

Fig. 1: Impact of Cost on the No. of Controllers

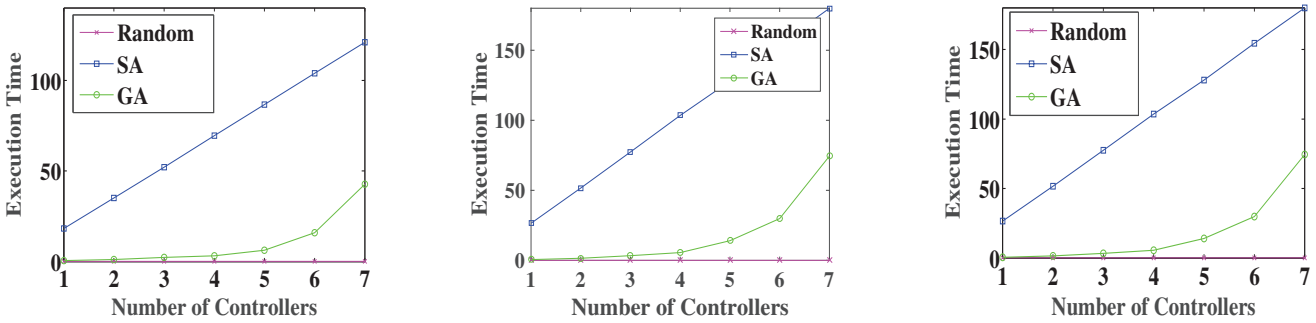

Fig. 2: Impact of Execution Time on the No. of Controllers

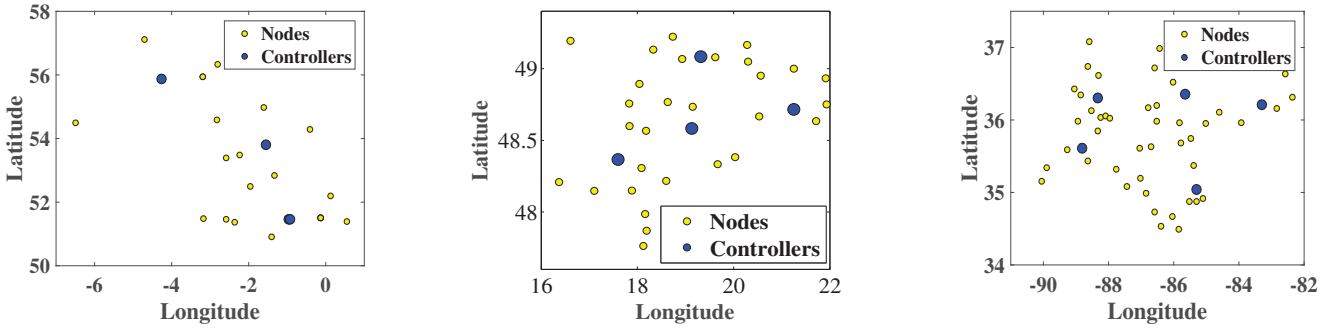

Fig. 3: Controller Locations using Proposed Genetic Algorithm

various datasets of the network. The experimental results show in favor of our proposed efficient genetic algorithm which highly outperforms the simulated annealing and random algorithm by giving very promising results. For our future work, we will consider some other performance metric such as resiliency and load balancing and apply it to some other real world topology.

ACKNOWLEDGMENT

This work is supported by the Department of Science and Technology under the WOS-A scheme with File No.SR/WOS-A/ET-133/2017.

REFERENCES

[1] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *IEEE communications surveys & tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014.

[2] K. S. Sahoo, S. Mohanty, M. Tiwary, B. K. Mishra, and B. Sahoo, "A comprehensive tutorial on software defined network: The driving force for the future internet technology," in *Proceedings of the International Conference on Advances in Information Communication Technology & Computing*. ACM, 2016, p. 114.

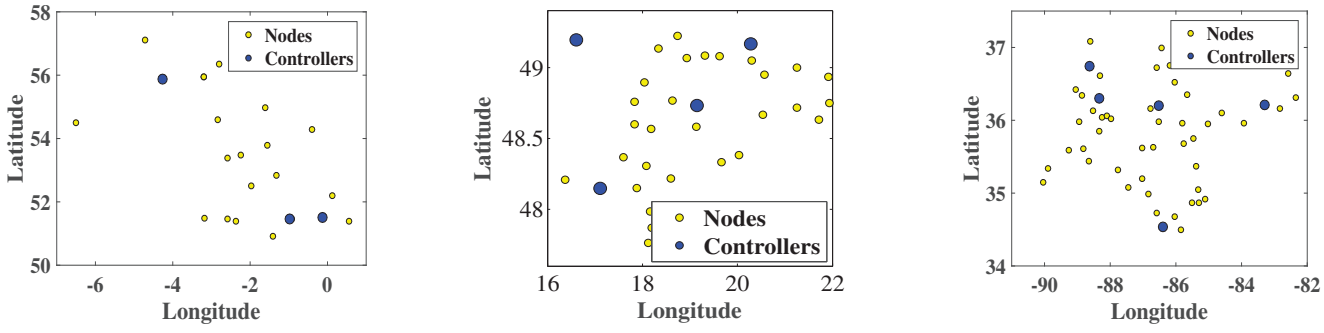

Fig. 4: Controller Locations using Simulated Annealing Algorithm

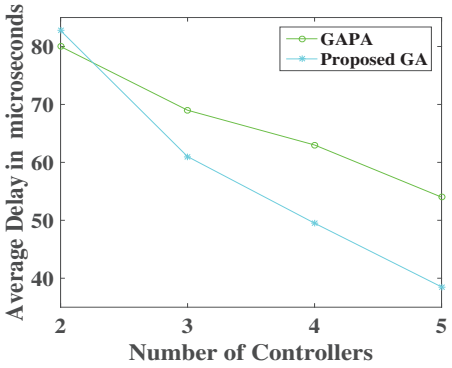

Fig. 5: Average Propagation Delay with No. of Controllers

[3] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 7–12.

[4] M. F. Bari, A. R. Roy, S. R. Chowdhury, Q. Zhang, M. F. Zhani, R. Ahmed, and R. Boutaba, "Dynamic controller provisioning in software defined networks." in *CNSM*, 2013, pp. 18–25.

[5] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 18, no. 8, pp. 1339–1342, 2014.

[6] A. Sallahi and M. St-Hilaire, "Optimal model for the controller placement problem in software defined networks," *IEEE communications letters*, vol. 19, no. 1, pp. 30–33, 2014.

[7] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale sdn networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 4–17, 2015.

[8] F. J. Ros and P. M. Ruiz, "On reliable controller placements in software-defined networks," *Computer Communications*, vol. 77, pp. 41–51, 2016.

[9] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-optimal resilient controller placement in sdn-based core networks," in *Proceedings of the 2013 25th International Teletraffic Congress (ITC)*. IEEE, 2013, pp. 1–9.

[10] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, "Reliability-aware controller placement for software-defined networks," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, 2013, pp. 672–675.

[11] B. P. R. Killi and S. V. Rao, "Capacitated next controller placement in software defined networks," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 514–527, 2017.

[12] A. Jalili, V. Ahmadi, M. Keshtgari, and M. Kazemi, "Controller placement in software-defined wan using multi objective genetic algorithm," in *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*. IEEE, 2015, pp. 656–662.

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

[14] H. Bo, W. Youke, W. Chuan'an, and W. Ying, "The controller placement problem for software-defined networks," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*. IEEE, 2016, pp. 2435–2439.

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

[16] V. Huang, G. Chen, Q. Fu, and E. Wen, "Optimizing controller placement for software-defined networks," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 224–232.

[17] O. Alp, E. Erkut, and Z. Drezner, "An efficient genetic algorithm for the p-median problem," *Annals of Operations research*, vol. 122, no. 1-4, pp. 21–42, 2003.

[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[19] topology-zoo. [Online]. Available: http://www.topology-zoo.org/

[20] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based sdn controller placement and assignment with minimum weight matching," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.