*International Journal of Engineering & Technology*, 7 (3.4) (2018) 1-5

**International Journal of Engineering & Technology**

*Website: www.sciencepubco.com/index.php/IJET*

***Research paper***

# A Comparison of Algorithms for Controller Replacement in Software Defined Networking

**Anitson T T[1]*, Smitha Vinod[2]**

[1] *Department of Computer Science, Christ University, Bengaluru, India*
[2] *Department of Computer Science, Christ University, Bengaluru, India*
**Corresponding author E-mail:anitsonttsat@gmail.com*

**Abstract**

Software Defined Network (SDN) partitions the control plane and the information plane to decrease the cost and for increasing the capacity for upgrading, and this will be helpful for the network administrators to manage the network services. It's the location where the controller exists. One of the main issue in Software Defined Networking is the location of the controllers as this could affect network execution and cost. In this paper, we have done an analysis on some algorithms that have been used for minimizing the number of controllers to reduce the latency, delay, etc. while placing a controller with the consideration of communication among the controller and the nodes.

***Keywords***: *Controller Placement, SDN controller, Greedy-SA algorithm, Network Partition, K-Critical algorithm, Latency, Software Defined Networking (SDN).*

## 1. Introduction

The principle behind Software Defined Networking is based on the partitioning of control plane (Control Layer) and the data plane (Infrastructure Layer) for reducing the cost and to increase the capacity to enhance [1].

In the figure (Figure 1) given below, three layers such as Control Layer, Application Layer and Infrastructure Layer are shown. Application layer contains Business Applications, that can be directly accessed by the end users and its where the business logics and improvement process happens. Control Layer is where all the automated data-driven fulfilment and assurance operations has been done. And the Infrastructure Layer is where all the data collection and machine learning analytics should be done.

SDN controller is one of the application used in Software Defined Networking for enabling the intelligent networking by managing the flow control. OpenFlow is one of the protocol that is used by SDN controllers, that enable servers to inform switches where the packets are to be transmitted. The controller lies between network devices and applications and it's the core of an SDN network. Any kind of communication between the application and the device has to be done with the help of controller. OpenFlow protocol is used by the controller for configuring the network devices and selecting the optimal network path for the application traffic [2].

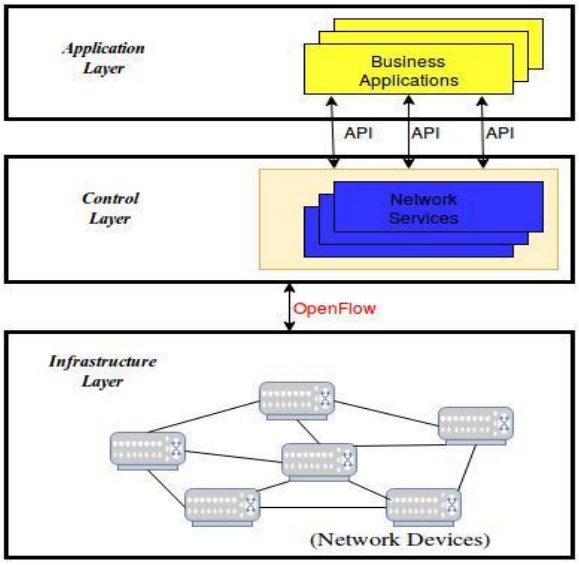


**Fig 1:** SDN Architecture

In this paper, we are doing a comparative study on some algorithms which are used for the Controller Replacement Problems, mainly on Greedy-SA algorithm and K-Critical algorithm based on the controller selection.

The paper is been followed as, in Section 2 discusses the various works related to controller placement problems. Section 3 discusses the proposed model based on the selected algorithms. Section 4, followed by comparison of the algorithms according to the results that has been collected. Section 5 gives the conclusion.

**Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.**

## 2. Related work

In [3], M. Huang et al. have developed an algorithm, Adaptive Bacterial Foraging Optimization (ABFO) for the calculation of complexity with the actual network state. According to this proposal the outcomes demonstrate this proposal, the outcomes can possibly manage optimization goals proficiently and successfully.

In [4], Y. Zhao et al. have proposed a K-means algorithm which is optimized for the investigation on the multi-controller placement issue from the point of view of reducing the latency. The network partition procedure has been introduced to reduce the issue. In particular, the network partition issue or problem and the controller placement issue are figured out initially. Based on the simulations conducted the results show that the proposed algorithm comparatively works surprisingly better than the standard K-means in minimizing latency between centroid and other nodes.

In [5], N. Jena et al. have proposed and implemented Simulated Annealing algorithm along with the Greedy heuristic to overcome the controller replacement problem. A network is given by a set of switches or routers. The controllers should be able to manage them. The algorithm finds the number of controller(s) required to cover all the network elements within the network in an optimal way. The primary criteria is the distance between all nodes and selected controllers is minimized. Controller's capacity is a constraint of the controller, that restricts a controller to manage an unlimited number of data plane devices.

In [6], L. Han et al. have proposed an algorithm which optimizes minimum-control-latency, based on the Greedy controlling pattern design. The thoughts and process are illustrated the Internet2 OS3Etopology comparison with the average-latency-optimized placement and lowest-case-optimized placement. Furthermore, a novel strategy in view of dynamic latency estimation is additionally displayed to get more exact control inactivity. Broad analyses have demonstrated that their minimum-controllatency-optimized can enhance the inequality when dividing software defined networking spaces and achieving the highest number of nodes per controller controlled.

In [7], L. Yao et al. have defined yet another metric to place controller, considering the weight of the node for an individual domain to start with, and after that they proposed a dynamic switch migration algorithm to adjust the flow of elements and acknowledge controller load balance in different Software Defined Networking areas.

In [8], G. Jourjon et al. proposed, LiDy, an algorithm which combines dynamic flow management and controller placement, and the evaluation for the controller utilization and latency on sparse and dense regions. The analysis shows that, LiDy is able to achieve a higher utilization than the latest controller replacement solution, in all settings.

In [9], Y. Hu et al. by considering the high complexity of the BIP in huge networks, a genetic heuristic algorithm is been proposed to locate a successful problematic arrangement. Simulation result demonstrate that the energy consumption of the heuristic algorithm is near that of the BIP solution. Close to 4%, extra connections are utilized by the heuristic algorithm if every one of the connections have a similar energy consumption.

In [10], C. Cervelló-Pastor et al. have proposed an algorithm called k-Critical for solving the controller placement problem in software defined networking. It will be finding the minimum number of controllers to fulfill an objective of intercommunication delay amongst controllers and nodes, and a management architecture will be created with those selected controllers that can improve the underlying network performance. They are concentrating on the controller selection method and demonstrate the desired management particularities that upgrade the control and management of the network. The outcomes demonstrate that their management trees adjust the heap among them and the data loss is been reduced.

## 3. Proposed model

### 3.1 Controller Selection Using K-Critical Algorithm

The k-Critical algorithm is one of the algorithm that has been used for illuminating the controller placement problem in software defined networking. By using this algorithm we can reduce the number of controllers required to cover the whole network and to select the controllers that authorize to construct robust and homogeneous management trees.

Controller placement relies upon the requirements and characteristics of the physical network. For choosing the best placement for the controller, the characteristics of the nodes are calculated regarding the network topology through the function θ which can be used for evaluating the candidate nodes, because of their location, to fulfil the delay '$\mathbf{D_{req}}$' to nodes that are not handled by a controller before. Along with that, they ensure that the resultant tree branches will have the most extreme delay amongst nodes and the controller, constrained by $\mathbf{D_{req.}}$

$$Function\theta = \gamma \times (connectivityofnode) + (1-\gamma) \times (weightofpath)$$

The coefficient γ measures the connectivity of a node and the weight of the path considering the candidate nodes. It is defined as,

$$\gamma = \frac{L_h}{L_{max}} \quad (1)$$

Where $0 \leq \gamma \leq 1$ And $\mathbf{L_h}$, is the greatest path length measured in number of jumps for the candidate node, and $\mathbf{L_{max}}$, is been defined as the greatest path length found among the candidate nodes. The connectivity of node relates the level of the candidate node to the node partition in its branches in i hops. The weight of the path relates the most extreme delay value that the candidate node achieves when it is the root, regarding Eq.1.

NB: γ is candidate node's least value with the smallest width, and 1 for the candidate nodes with the biggest width.

Algorithm for the Controller Selection using K-Critical are as follows:

Required: $(NxN)$ Speed Delay Matrix. $Dreq \leftarrow$ Required.Delay

*CT* = Set of candidate nodes that satisfies the $Dreq$

If $CT \neq 0$

Then

For every node $nd \in CT$ has be converted to a controller

Go for

Evaluating θ

End on

Selecting the node $nd \in CT$ with the maximum values in θ; count =0.
Else

While there are no more nodes associated with the cluster

Go for

$C_{nd}$ ← Finding the outermost node to the cluster

$C_{cnd}$ ← Finding the set of the candidate nodes for managing the cluster
For all the candidate node $nd \in Ccnd$

Go for

Evaluation θ

End for

Selecting the node nd which is having maximum value in θ

$Cluster_k$ ← Finding from $(NxN)$ the nodes *vnd* that satisfies $d(vnd, nd) \leqslant Dreq$, Where the $vnd \notin cluster$

$$cluster \leftarrow cluster \cup cluster_k, CT = CT \cup nd, cluster_k \leftarrow 0, C_{nd} \leftarrow 0$$

End all

End if

### 3.2 Controller Selection using Greedy-SA algorithm

The Greedy-SA algorithm is a combination of both Greedy algorithm and SA algorithm that has been used for the problems in case of controller replacement. Basically the SA (Simulated Annealing) algorithm is used for solving the optimization problems. The SA technique they have used for controller selection is by considering N number of nodes which will be considered as controllers for the network, with the help of a method in which the number of ones in the string suggests that the quantity of controller has been picked, though a zero infers that the relating network component isn't a controller, but instead it will be serving as a switch. The SA will be looking exactly for the N ones among them. The standard SA algorithm won't be able to deal with the fixed set of ones among the string, so we have to add an additional opeartor(oper) to the algorithm. The additional operator will be acting like as follows:

$$oper(x) = \begin{cases} if(k < N)then, place(N-k)numberofonesrandomly \\ otherwise, removethe(k-N)numberofonesramdomly \end{cases}$$

The standard SA algorithm is used along with the Greedy algorithm and while finding an answer using Greedy algorithm, it goes for the best immediate and starts with a permutation of,

$pi(M_{N\_K})$

Along with the Euclidean distance among switch 'i' and controller 'j' is been calculated with,

$C_l \leftarrow Cn_{ij}$

The nodes will be assigned to the nearest controller. In this situation the controller don't have the ability to deal with the node, the algorithm inquires for the accompanying closest controller and do a similar operation. This procedure will proceed until the controller could find and designate rest of the node to the controller. The greedy solution given by the algorithm goes wrong when no controller can suit the required limit, that case is considered as the more terrible case.

Greedy Heuristic Algorithm used for the Controller Selection is:

For every Binary string S of SA(Simulated Annealing)

Pick N number of one as controllers

Pick M - N number of zero as the nodes

Select a permutation $pi(M_{N\_K})$ at arbitrary

For each terminal $pi(M_i)$

Go for

Determining $C_{ij}$ = separate from $pi(M_i)$ to the nearest Controller $Cn_j$

$Cn_j \leftarrow pi(M_i)$

$C_l \leftarrow Cn_{ij}$

Finding F (z) = sum of ($C_l$)

end on

## 4. Evaluation and Results

All K-Critical algorithm have been used to demonstrate the assumption on the controller selection in the controller performance regarding delay, node management allotment, data loss among controllers and depth of the tree. To facilitate this, three system classification are created, i.e. dense, medium and sparse networks. Software defined in [11] is used to create these networks. At that point they assessed the performance and strength of the trees built at the controllers chosen and for every single one of the exhibited answers for every last one of the network levels.

The controllers were found for various delay values in the scope of microseconds. The outcomes displayed were the normal outcomes got for each system class assessed. MATLAB was used for calculating the algorithm and their performance.

The calculation of the data loss can be done by,

$$d_i = \frac{1}{N-1} \sum_i n_i \qquad (2)$$

Where, $i \in V(T_k)$.

By considering data loss '$d_i$' for a node, i relies upon the number of the nodes '$n_i$' that are established, and on the probability that the node i goes down. Along with the consideration of each node in the tree topology '$T_k$' has a similar loss probability and 'N' will be the number of nodes in the network while '$V(T_k)$ will be the nodes in the tree '$T_k$'.

One of the result says that for the medium and the sparse connectivity networks, the controllers chosen by K-Critical have preferred the node allotment over alternate solutions. The outcomes acquired for dense networks isn't appeared, on the grounds that only one controller deals with the entire network for all cases.

The next result says that the depth of the tree is been influenced by the allotment of node for each controller. Those trees made out of controllers found by K-Median is having the most exceedingly bad node appropriation with lengthy paths is an example. For the dense network, whereas just one controller is needed, K-Critical will be allotting the best node among the branches, therefore the most limited ones are the subsequent trees.

So the best controller allotment can be done by the K-Critical. That best outcomes acquired utilizing K-Critical are because of the measure to choose the leading controller. That determination relies upon the bad node in the network in form of delay, the demanding node. Then again, k-Median doesn't achieve better performance as a result of the foremost criteria to choose the controller isn't productive.

The computational complexity for every single controller placement methods assessed by O(nk). While locating a controller it requires checking the least space in the network among the N nodes. So, the computational complexity for locating a controller also can be characterized by O(n).That mechanism has to be repeated till the k controllers were found.

By using Greedy-SA algorithm, the result says that the average delay got by Greedy – SA algorithm is consistently stable. In the same way the results on average latency says that, while the quantity of the controllers are continuously expanding under a similar topology in every technique [12].

while evaluating the computational time with expanding controllers in a given topology while the size of the network is the same. It has seen that, by extending more number of controllers, the confluence of time is bigger. At the point when the number of controllers expands the growth rate of calculation also raise.

## 5. Conclusion

In this paper the algorithms that has been discussed are mainly used for the controller selection process along with the assessment based on the involvement of the controller selection in the succeeding control layer. The demonstration includes the number of controllers that's been selected along with their locations for determining the performance of the network, and the control network robustness controller selection can also be influenced significantly because of bad selection of controller, as well as the network operation also. According to the results obtained the best count on the controllers are based on the components of physical network along with the specifications, an excellent selection of controllers will enable the network while adjusting the weight as well.

By using k-Critical for choosing the controllers a strong control layer in suspense of network interruptions is developed. This is because of the physical components of the network are considered in the time of selecting appropriate controllers and the finest location for the controller can also be organized meanwhile a network has to be enlarged or by replacing the load.

One of the major challenge in the designing of the control plane in architecture of SDN. Brute force access with this issue is basically attainable for small and medium size network, so another mechanism is required for large example of networks. Typically heuristic approach includes to meet the time and resource demand. According to the results obtained by using Greedy-Sa algorithm, the number of controllers in the control plane has been known formerly, however their positions majorly affect numerous ongoing issues ([12] [13]).

In future work the arrival and execution of real time task as other important metric for the said problem and implement the cost function on a real time environment can be considered. Along with defining how the tree topology is built from the selected controllers taking into account several network performance metrics. This is because selecting the shortest paths between controller-nodes is not the best option for improving the load and the robustness of the control layer.

## References

[1] Wang, Guodong, Yanxiao Zhao, Jun Huang,Qiang Duan, and Jun Li. "A K-means-based network partition algorithm for controller placement in software defined network", 2016 IEEE International Conference on Communications (ICC), 2016.

[2] Afrim Sallahi, Marc St-Hilaire. "Expansion Model for the Controller Placement Problem in Software Defined Networks", IEEE Communications Letters, 2016.

[3] B. Zhang, X. Wang, L. Ma and M. Huang, "Optimal Controller Placement Problem in Internet-Oriented Software Defined Network," 2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Chengdu, 2016, pp. 481-488.

[4] 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," 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016,pp. 1-6.

[5] K. S. Sahoo, B. Sahoo, R. Dash and N. Jena, "Optimal controller selection in Software Defined Network using a greedy-SA algorithm," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom),New Delhi, 2016, pp. 2342-2346.

[6] L. Han, Z. Li, W. Liu, K. Dai and W. Qu, "Minimum Control Latency of SDN Controller Placement," 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, 2016, pp. 2175-2180.

[7] L. Yao, P. Hong, W. Zhang, J. Li and D. Ni, "Controller placement and flow based dynamic management problem towards SDN," 2015 IEEE International Conference on Communication Workshop (ICCW), London, 2015, pp. 363-368.

[8] M. T. I. ul Huque, G. Jourjon and V. Gramoli, "Revisiting the controller placement problem," 2015 IEEE 40th Conference on Local Computer Networks (LCN),Clearwater Beach, FL, 2015, pp. 450-453.

[9] Y. Hu, T. Luo, N. C. Beaulieu and C. Deng, "The Energy-Aware Controller Placement Problem in Software Defined Networks," in IEEE Communications Letters, vol. 21, no. 4, pp. 741-744, April 2017.

[10] Y. Jiménez, C. Cervelló-Pastor and A. J. García, "Defining a network management architecture," 2013 21st IEEE International Conference on Network Protocols (ICNP), Goettingen, 2013, pp. 1-3.

[11] M. Bastian, S. Heymann, and M. Jacomy Gephi: an open source software for exploring and manipulating networks,Proceedings of International AAAI Conference on Weblogs and Social Media (ICWSM).,pp. 361-362. California, USA, 2009.

[12] Sahoo, Sampa, et al. "Execution of real time task on cloud environment." (2015).

[13] Kumar, Dilip, and Bibhudatta Sahoo. "Energy efficient heuristic resource allocation for cloud computing." (2014).