2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress

2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics) | 978-1-6654-5417-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics55523.2022.00074

A Density-based Controller Placement Algorithm for Software Defined Networks

1st Jue Chen
Shanghai University of Engineering Science
Shanghai, China
jadeschen@sues.edu.cn

2nd Yujie Xiong
Shanghai University of Engineering Science
Shanghai, China
xiong@sues.edu.cn

3rd Dun He
Shanghai University of Engineering Science
Shanghai, China
1273562746@qq.com

***Abstract*—Software Defined Network (SDN) is a new type of network structure, which decouples the data plane from the control planes, and improves network flexibility. However, this separation creates a problem called Controller Placement Problem (CPP), that is, how many controllers are required and where they should be placed. In this paper, we propose an improved Density-based Controller Placement Algorithm (DCPA), which can obtain the required number of controllers through traversing candidate values of radius, and then divide the entire network into multiple sub-networks. In each sub-network, the controllers are deployed with the purpose of minimizing the average propagation latency and worst-case propagation latency between controllers and switches simultaneously. We conduct experiments on 100 real network topologies from the Internet Topology Zoo to evaluate the performance of algorithm, and the results verify that DCPA can always find out the controller placement scheme with a low time consumption to reduce the propagation latency for different network scales, with a less than 10% margin from the optimal solution.**

***Index Terms*—SDN, Controller Placement Problem, Density Based Clustering, Propagation Latency**

I. INTRODUCTION

As a new network architecture, Software Defined Network (SDN) separates the control plane from the data plane (as shown in Figure 1), which not only enables the independent evolution of two planes and helps to solve the problem of network rigidity [1–3], but also brings multiple advantages that traditional network architecture does not have, such as programmable capabilities, centralized control, and simplified network management. All of these features make SDN widely be concerned by academia and industry.

With the expansion of the SDN network, it is hard for a single controller to manage all switches, as the capacity of the controller is limited, which may cause the propagation latency between the controller and switches to become much larger and then affect the performance of the whole network. In addition, the controller is also prone to single point of failure [4]. In order to improve scalability, security and elasticity of SDN, it is necessary to place multiple controllers in the network. What followed is the Controller Placement Problem (CPP) [5], which is formulated to quantify the performance of the control plane and mainly considers three issues: (1) the number of controllers, (2) the location of controllers, and (3) the allocation between controllers and switches.

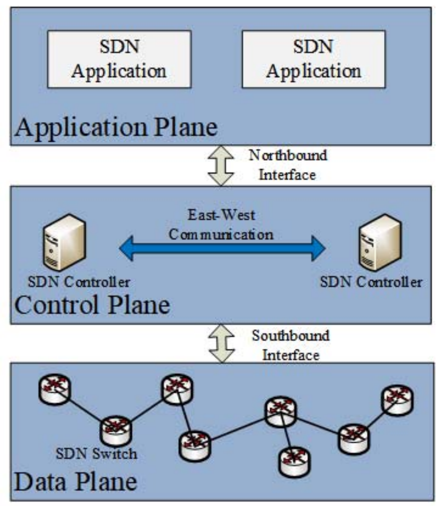

Fig. 1. SDN Architecture

In this paper, we propose an improved Density-based Controller Placement Algorithm (DCPA) to solve CPP, including calculating the required number of controllers and minimizing both average propagation latency and worst-case propagation latency. Firstly, the algorithm traverses the possible values of the radius and determines the required number of controllers according to the principle of clustering [6]. Secondly, we minimize both average propagation latency and worst-case propagation latency simultaneously through the calculated number of controllers. Thirdly, after calculating the density and weight for all nodes, the node with the highest weight is set as the first initial cluster center. According to the previous cluster center and the weight of the remaining nodes in the network topology, we find the next initial cluster center, and this process is repeated until the number of controllers is obtained. Finally, we determine the best locations for controllers by moving the candidate locations in each sub-network, and connect all switches to the nearest controllers to construct the whole SDN network through the K-means algorithm. Generally, DCPA has fast response speed with stable solution, which is easy to apply in real networks.

978-1-6654-5417-9/22/$31.00 ©2022 IEEE

DOI 10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics55523.2022.00074

The rest of this article is organized as follows. The third part investigates related work. The fourth part describes the proposed new solution in detail. The fifth part presents the performance analysis based on DCPA. The sixth part is a summary of this article.

II. Related Work

Controller placement problem in SDN is a critical issue that has attracted great attention in the literature. The main purpose of CPP is to identify the best locations to place the controllers to improve the performance of SDN network [7]. According to the optimization targets, existing research works can be roughly divided into four categories: latency, cost-efficiency, resilience and reliability, and multi-objective [8, 9]. Since all the functions of the network are performed through the message exchange between controllers and switches, the latency between the control plane and the data plane seriously affects the performance of the network. As a result, this paper focuses on minimizing the propagation latency of SDN.

CPP is first proposed by Heller et al. [10]. Their main works are to investigate the influence of controller placement on propagation latency (including average propagation latency and worst-case propagation latency). Moreover, CPP is formulated as a facility location problem and K-center is adopted to solve this problem. Wang et al. [11] propose a network partition algorithm based on K-means, which divides the entire network into multiple sub-networks according to the physical distance, so as to decrease the latency between the controller and switches in each sub-network. Han et al. [12] propose an algorithm based on greedy strategy that improves the imbalance when partitioning the network and achieves the maximum number of nodes controlled by each controller. Yao et al. [13] regard both the capacity of controllers and latency as optimization goals at the same time, and turn CPP into a K-center problem with capacity constraints.

According to our investigations, most of the above-mentioned works are done under a given number of controllers. However, it may be impossible for us to know how many controllers need to be placed in the SDN in advance. Even though the optimal number of controllers can be obtained through the traversal method, it is obviously not advisable in a large-scale network. Therefore, in this paper, we propose a new fast clustering algorithm to solve the problem of how many controllers are needed and where controllers should be deployed, considering simultaneously minimizing the average propagation latency and worst-case propagation latency.

III. Algorithm

In this section, an improved Density-based Controller Placement Algorithm (DCPA) [14–16] is proposed to address CPP for SDN. The objective of DCPA is to minimize propagation latency between switches and controllers after the required number of controllers has been calculated.

In DCPA, we first calculate the required number of controllers for the SDN network through traversing all the candidate values of density radius. After the optimal density radius is decided (which will be introduced in the next section), we perform the SDN network partition strategy based on following three indicators: the node density, the intra-cluster aggregation degree and the inter-cluster separation degree. Secondly, we traverse all candidate locations in each sub-network to minimize the average propagation latency and the worst-case propagation latency. The parameters used in the DCPA algorithm and definitions of these indicators are given as follows.

After determining the optimal density radius d_c, the following three indicators are calculated. The node density ρ of a node is defined as the number of nodes adjacent to the current node with the distance less than or equal to d_c.

$$\rho_i = \sum_{j=1}^{N} \lambda(d_c - d_{ij}) \tag{1}$$

$$\lambda(x) = \begin{cases} 1, & x \geq 0; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In equation (1), d_{ij} represents the distance switches v_i and v_j. The d_c is a threshold distance, and only the switch with the distance below or equal to d_c is considered as a neighbour of the current node. The nodes in the same circle are grouped into a cluster and we calculate the average propagation distance between these nodes in the same cluster, which is called the intra-cluster aggregation degree σ_i. We define it as follows:

$$\sigma_i = \frac{2}{\rho_i\left(\rho_i - 1\right)} \sum_{i=1}^{\rho_i} \sum_{j=i+1}^{\rho_i} d_{ij} \tag{3}$$

The inter-cluster separation degree δ_i represents the propagation distance between node i and another node j with a higher node density. If node i does not have the maximum density, δ_i is defined as $min(d_{ij})$. On the contrary, if node i has the maximum density, and then δ_i is defined as $max(d_{ij})$. The mathematical expression is described as follows:

$$\delta_i = \begin{cases} min(d_{ij}), & \rho_i < \rho_j; \\ max(d_{ij}), & \text{otherwise.} \end{cases} \tag{4}$$

Based on three indicators mentioned above, the weight of each node is defined as follows:

$$\omega_i = \rho_i \times \delta_i \times \frac{1}{\sigma_i}, \quad \forall i \in N \tag{5}$$

According to equation (5), the node with the largest weight ω in the SDN network is selected as the first cluster center. We introduce the parameter Γ to select the remaining controller positions, so as to achieve SDN network partition:

$$\Gamma_j = \omega_j \cdot d(j, s_{i-1}), \quad \forall j \in v_i \tag{6}$$

$$s_i = \max_{j \in v_i} \Gamma_j \tag{7}$$

Note that the nodes adjacent to the first controller node with the distance less than or equal to d_c will not participate in the selection of the following controllers. In equation (6), $d(j, s_{i-1})$ is the propagation distance between the next controller to be selected and the node that has been selected as the controller last time. It can be inferred from equation (7) that the position of the next controller needs to meet the characteristics of both large weight ω and long propagation distance $d(j, s_{i-1})$. This screening method also reflects that the nodes to be chosen as controllers are subject to the principle of high distribution. As a result, the controller positions are determined through a K-means algorithm, which reduces the number of iterations and time complexity.

The algorithm of DCPA is shown in Algorithm 1, which realizes the optimization of both average propagation latency and worst-case propagation latency. In steps 1-14, we partition the entire network with the strategy that promises the high intra-cluster aggregation degree and high inter-cluster separation degree. In addition, placing one controller in each cluster can optimize the propagation latency as much as possible. In steps 15-18, DCPA finds the best locations for controllers through K-means algorithm, and select the best controller locations in each cluster by traversing all candidate locations with the goal of minimizing the worst-case propagation latency.

Algorithm 1 Density-based Controller Placement Algorithm (DCPA)

Input: $G = (V, E), k, d_c, W, first\ node$
Output: Clustering results

1: $S = \phi$;
2: $select\ first\ node\ with\ d_c\ as\ cluster\ S_1$;
3: $remove\ cluster\ S_1\ from\ V$;
4: $renew\ V$;
5: $j = 2$;
6: **while** $j\ <= k$ **do**
7:     **for** each $i \in V$ **do**
8:         $S_j = Select\ next\ cluster$ $center(i, C_{j-1}, d_c, \Gamma_i, W, G)$;
9:     **end for**
10:     $remove\ cluster\ S_j\ from\ V$;
11:     $renew\ V$;
12:     $S+ = S_j$;
13:     $j+ = 1$;
14: **end while**
15: $use\ the\ set\ S\ to\ execute\ a\ standard\ K-mea-$ $ns\ algorithm\ to\ partition\ to\ generate\ SDN_i$ $(V_i, E_i), \quad \forall i \in k$;
16: $perform\ a\ traversal\ for\ each\ node\ in\ each\ SDN$ $partition\ and\ use\ min(L_{worst}(S))\ as\ the\ principle$ $to\ determine\ the\ final\ controller\ locations$;
17: $renew\ S$;
18: $Calculate\ SDN_i(V_i, E_i), \quad \forall i \in k$

IV. Performance Evaluation

To prove the effectiveness of our proposed algorithm, we compare the performance between DCPA and the brute force algorithm which can always promise the optimal solution. As the number of controllers is affected by the value of density radius, we firstly traverse all the candidate values of density radius to find out the optimal solution. In the experiments, we choose totally 50 topologies from the Internet Topology Zoo to compose the training set to make the simulation results more convinced. In each topology, the candidate values of density radius contain 0.05, 0.1, 0.2, 0.3, 0.4 and 0.5. In terms of each value, we perform both DCPA and brute force algorithm to calculate the controller placement scheme and the corresponding propagation latency, including the average number of hops and the maximum number of hops between controllers and switches, as shown in Figure 2. It can be observed from the figure that when the density radius equals to 0.05, the error rate is less than 10% for most situations. When the values of density radius increase, the error rate fluctuates. Especially when the value equals to 0.4, the worst situation can be observed and the error rate is even more than 60%. As a result, we set the density radius to be 0.05 in the subsequent experiments.

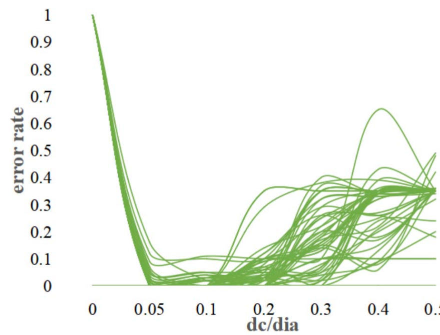

Fig. 2. Error Rate between DCPA and Brute Force with Different Values of Density Radius

In the testing set, we choose another 50 topologies from the Internet Topology Zoo to verify the effectiveness of DCPA, and the network scales increase from 20 nodes to 160 nodes. The simulation results are shown in Figure 3. It can be observed from the figure that when density radius is set to be 0.05, the error rate between DCPA and the brute force is always less than 4%, which is acceptable.

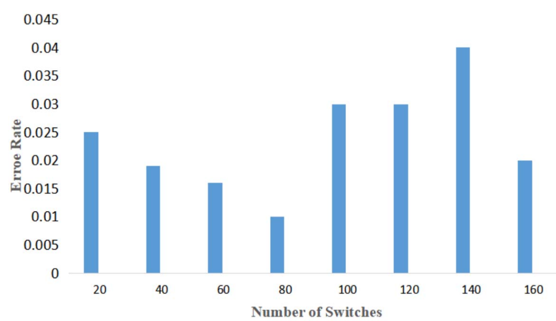

Fig. 3. Error Rate between DCPA and Brute Force with Different Network Scales

On the other hand, the time consumption of DCPA can be divided into 5 parts: calculating the node density, calculating the intra-cluster aggregation degree, calculating the inter-cluster separation degree, partitioning the network and finding best locations to place controllers. Firstly, the time consumption of calculating the node density is greatly influenced by the size of the network topology. The more switches in the network, the longer the search time required, and the time complexity of this part can be calculated as $O(\bar{\rho}N)$. Secondly, the time complexity of computing the intra-cluster aggregation degree, calculating the inter-cluster separation degree, and partitioning SDN network is $O(\bar{\rho}N)$, $O(N)$ and $O(N-\bar{\rho})$, respectively. Finally, in order to find best locations to place controllers, we need to traverse all nodes in each sub-network, so that the time complexity of last part is $O(\frac{N^2}{k})$. Therefore, the total time complexity is $O(\frac{N^2}{k}+\bar{\rho}N+\bar{\rho}N+N+N-\bar{\rho})$. On the contrary, the time complexity of brute force algorithm is $O(N^k)$. As a result, it can be derived that DCPA can always promise an optimized controller placement scheme, with a low time consumption and a less than 10% margin from the brute force algorithm.

V. Conclusion

In this paper, an improved Density-based Controller Placement Algorithm (DCPA) is applied to reduce the propagation latency between controllers and switches. Firstly, We calculate the required number of controllers through traversing all the candidate values of density radius, and calculate the weight for each network node. The weight is compose of three indicators including the node density, the intra-cluster aggregation degree and the inter-cluster separation degree, and the node with the largest weight is selected as the first initial cluster center. After removing the nodes within the optimal radius of the first center, we choose the node with the maximum value as the next initial controller position. Repeat the above process until the number of partitions reaches the optimal number of controllers. In the end, we determine the final placement of each controller in each sub-network through the K-means algorithm. In order to evaluate the performance of the proposed algorithm, extensive simulations are conducted under 100 real topologies from the Internet Topology Zoo. Simulation results verify that DCPA can always find out the optimal solution with a low time consumption for placing controllers for different network scales, and with a less than 10% margin from the brute force algorithm. In future works, the propagation latency calculated through the actual physical distance is regarded as our optimization goal.

VI. Acknowledgement

This work was supported by Science and Technology Commission of Shanghai Municipality (STCSM) under grants No.19YF1418300 and No.19YF1418400.

References

[1] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider. NFV and SDN - Key Technology Enablers for 5G Networks. *IEEE Journal on Selected Areas in Communications*, PP(11):1–1, 2018.

[2] F. Hu, Q. Hao, and K. Bao. A survey on software-defined network and openflow: From concept to implementation. *Communications Surveys & Tutorials IEEE*, 16(4):2181–2206, 2014.

[3] D Kreutz, Fmv Ramos, P. E. Verissimo, CE Rothenberg, and S. Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1), 2014.

[4] Zehua Guo, Ruoyan Liu, Yang Xu, Andrey Gushchin, Anwar Walid, and H Jonathan Chao. Star: Preventing flow-table overflow in software-defined networks. *Computer Networks*, 125:15–25, 2017.

[5] Tamal Das, Vignesh Sridharan, and Mohan Gurusamy. A survey on controller placement in sdn. *IEEE communications surveys & tutorials*, 22(1):472–503, 2019.

[6] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *science*, 344(6191):1492–1496, 2014.

[7] Jue Chen, Yu-Jie Xiong, Xihe Qiu, Dun He, Hanmin Yin, and Changwei Xiao. A cross entropy based approach to minimum propagation latency for controller placement in software defined network. *Computer Communications*, 2022.

[8] Junlong Zhou, Jin Sun, Mingyue Zhang, and Yue Ma. Dependable scheduling for real-time workflows on cyber-physical cloud systems. *IEEE Transactions on Industrial Informatics*, 17(11):7820–7829, 2021.

[9] Junlong Zhou, Kun Cao, Xiumin Zhou, Mingsong Chen, Tongquan Wei, and Shiyan Hu. Throughput-conscious energy allocation and reliability-aware task assignment for renewable powered in-situ server systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(3):516–529, 2022.

[10] Brandon Heller, Rob Sherwood, and Nick McKeown. The controller placement problem. *ACM SIGCOMM Computer Communication Review*, 42(4):473–478, 2012.

[11] Guodong Wang, Yanxiao Zhao, Jun Huang, Qiang Duan, and Jun Li. A k-means-based network partition algorithm for controller placement in software defined network. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[12] Lin Han, Zhiyang Li, Weijiang Liu, Ke Dai, and Wenyu Qu. Minimum control latency of SDN controller placement. In *2016 IEEE Trustcom/BigDataSE/ISPA*, pages 2175–2180. IEEE, 2016.

[13] Guang Yao, Jun Bi, Yuliang Li, and Luyi Guo. On the capacitated controller placement problem in software defined networks. *IEEE Communications Letters*, 18(8):1339–1342, 2014.

[14] Hiromasa Kaneko and Kimito Funatsu. Data density-based fault detection and diagnosis with nonlinearities between variables and multimodal data distributions. *Chemometrics and Intelligent Laboratory Systems*, 147:58–65, 2015.

[15] Yanfeng Zhang, Shimin Chen, and Ge Yu. Efficient distributed density peaks for clustering large data sets in mapreduce. *IEEE Transactions on Knowledge and Data*

Engineering, 28(12):3218–3230, 2016.
[16] Jianxin Liao, Haifeng Sun, Jingyu Wang, Qi Qi, Kai Li, and Tonghong Li. Density cluster based approach for controller placement problem in large-scale software defined networkings. *Computer Networks*, 112:24–35, 2017.