Received September 20, 2019, accepted November 11, 2019, date of publication November 15, 2019, date of current version December 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2953723

Multi-Controller Placement for Load Balancing in SDWAN

KONGZHE YANG, DAOXING GUO, BANGNING ZHANG, AND BING ZHAO
Communications Engineering College, Army Engineering University of PLA, Nanjing 210007, China

Corresponding author: Bangning Zhang (bangning_zhang@sina.com)

This work was supported in part by the National Science Foundation of China under Grant 61401508 and Grant 61471392, and in part by the Natural Science Foundation of Jiangsu Province under Grant 20150719.

ABSTRACT The capability of flexible network management makes it available to incorporate software-defined networking (SDN) in the wide area network (WAN). Thereinto, multiple controllers are deployed in the software-defined wide area network (SDWAN) to tackle the performance bottleneck. Thus, how to effectively solve the load balancing problem in the control plane towards SDWAN under different constraints is worthy of further study. In order to achieve load balancing among the distributed controllers, we proposed a low-complexity controller placement algorithm, Simulated Annealing Partition-based K-Means (SAPKM), towards SDWAN. Meanwhile, two cost functions were proposed as the load balancing indices to assess the efficiency of the proposed algorithm from the perspective of topology structure and flow traffic distribution, respectively. Simulations were performed by using actual topologies on Topology Zoo with different sizes and different structures. Experimental results demonstrated the effectiveness of SAPKM in reducing both average load and load balancing indices with propagation latency and network reliability performance enhanced simultaneously.

INDEX TERMS Software-defined wide area network (SDWAN), controller placement, load balancing, propagation latency, network reliability.

I. INTRODUCTION

Software-Defined Networking (SDN) is a new paradigm which departs the control plane from the data plane to further improve network performance and manage the whole network more efficiently and flexibly. Multiple physically distributed controllers composing the control plane offer a logically centralized vision of the network and manage forwarding devices in the data plane to forward packets and to execute the handover. Large scale networks, e.g. the software-defined wide area network (SDWAN), need numerous dedicated controllers to meet performance metrics, including latency, reliability, etc. Thus, how to determine the number and the locations of the controllers with appropriate assignments for solving load balancing issue in SDWAN worth in-depth study.

Heller *et al.* [1] introduce the controller placement problem (CPP) for the first time. The authors study the impact of the number and locations of the controllers on both average/worst-case latencies and prove that the controller placement problem is NP-hard. In OpenFlow v.1.3, an SDN controller can be any role including the master, slave, and equal [2], [3]. Controllers can change their roles at any time which allows dynamic switch migration to maintain load balancing among distributed controllers. However, since the propagation latency dominates the performance and dynamic migration will definitely increase both flow setup time and response time, the relatively statical assignment is the normality of WANs.

Concerning WAN, Xiao *et al.* [4] firstly define the metrics of the CPP in WANs regarding both latency and reliability by incorporating spectral clustering to partition the domains inside. Considering heterogeneity and interconnections of controllers, Sallahi and St-Hilaire [5] study the optimal model for CPP, which is merely practical for small scale networks.

Heuristic toolset named POCO is adopted in [6] to address the CPP with respect to requirements range from latency constraints to load balancing and failure tolerance, which compensates the loss of accuracy for faster computation times. Zhang *et al.* [7] mainly focus on the control traffic

The associate editor coordinating the review of this manuscript and approving it for publication was Malik Jahan Khan.

exchanged among the controllers and formulate an analytical model to estimate the impact of inter-controller communications on the reaction time perceived at the switches. The authors also evaluate the trade-offs between two kinds of latencies by using real ISP network topologies for simulation. Wang *et al.* [8] analyze the end-to-end latency and queueing latency on controllers in detail, and partition the WANs into smaller domains by using and proving effectiveness of the centroied-based clustering algorithm in decreasing both kinds of latencies.

Speaking of network reliability, Hu *et al.* [9] address the CPP and firstly present the expected percentage of control path loss as a metric to characterize the reliability of given networks. The authors not only prove the NP-hardness of reliability-aware CPP but also validate the effectiveness of the simulated annealing algorithm (SA) on maximizing network reliability without introducing unacceptable latencies. Another controller placement strategy named Survivor introduced by Müller *et al.* [10] further improves the survivability of SDN networks by considering path diversity, capacity-awareness of controllers, and failover mechanisms during initial placement. Ros and Ruiz [11] firstly shows that the network reliability is principally determined by the network topology structure, especially the density of the internal nodes. The authors introduce the fault-tolerant CPP and propose a heuristic algorithm to provide solutions with required reliability, but they don't optimize the propagation latency. Besides, Jiménez *et al.* [12] build robust trees by using the proposed k-Critical algorithm to discover the minimum number and the locations of controllers for creating a robust control layer in response to network disturbances.

On the other hand, the dynamic controller placement problem (DCPP) is raised and defined in [13] for the first time. The authors propose a framework with multiple dedicated controllers working cooperatively according to network dynamics. However, the formulated optimal model and two corresponding heuristic algorithms neglect to evaluate the network reliability. In addition, ul Huque *et al.* [14] address the DCPP consisting of two parts, determining the number and locations of controllers to support a dynamic load and bound communication latencies, respectively. The authors evaluate the proposed algorithm named *LiDy*+ with the time complexity $O(n^2)$ on both sparse and dense networks.

Compared with SDWAN, the satellite networks [15] and integrated terrestrial-satellite networks [16] also have large coverage for the types of forwarding equipment in the data plane. Besides, both the satellite networks [17] and integrated terrestrial-satellite networks [18] subject to the limited and unbalanced network resources. How to explore new network architectures to supply services [19] and applications [20] with various QoS/QoE requirements in different scenarios [21] is challenging. For satellite communications towards 5G [22], Papa *et al.* [23] considered the DCPP in an LEO constellation satellite network for minimizing the average flow setup time.

The controller load balancing is another inevitable and crucial performance metric in the control plane of SDWAN. Dixit *et al.* [24] introduce an elastic distributed controller architecture to maintain controller load balance by changing the number of controllers according to traffic conditions. Ahmed and Boutaba [25] focus on the design considerations, e.g. flow setup and monitoring, for a multi-controller architecture towards SDWAN. Yao *et al.* [26] define a node weight for controllers and proposed a switch migration algorithm to realize controller load balance according to flow dynamics. Sallahi and St-Hilaire [27] introduce an expansion model minimizing update cost when re-organizing network topologies. Tingting *et al.* [28] regard the controllers and the switches as the elements of a bipartite graph and proposed a Kuhn-Munkres based minimum weight matching algorithm to decrease the propagation latency and load imbalance. Recently, Li *et al.* [29] propose a load balancing based dynamic multi-controller placement scheme, which considers the propagation latency and the controller capacity as the main factors for controller placement. The authors take both intra-domain and inter-domain communication costs as optimization targets, and propose two algorithms to deal with initial static and dynamic states, respectively. Hou *et al.* [30] introduce a multi-controller placement scheme in hierarchical architecture towards SDWAN. The proposed algorithm enhanced performance metrics to varying degrees, including load balancing, request latency, and control plane reliability.

Unlike the above-mentioned papers, we define two different cost functions in term of the network topology structure and flow traffic distribution to estimate load balancing efficiency, and hybridize the network partition scheme to solve the load balancing controller placement problem. The entire network is divided into multiple sub-domains, and only one dedicated controller is deployed in each of them. By incorporating the centroid-based clustering algorithm [8], we propose the Simulated Annealing Partition-based K-Means (SAPKM) algorithm to solve the load balancing controller placement problem. The main contributions of this paper are briefly highlighted as follows.

- The Simulated Annealing Partition-based K-Means (SAPKM) algorithm is proposed to ameliorate the controller placement problem in SDWAN for minimizing average controller load and enhance other performance metrics, including propagation latency and network reliability.
- Two weighted cost functions are defined as the load balancing indices in terms of network topology structure and flow traffic distribution, respectively. We not only consider the number of switches and the flow request rate managed by the controller in each sub-domain but also take the node degree of each switch and the propagation latency between the switches and corresponding controller at the same time.
- The decision conditions can be defined as average controller load, normalized weighted node degree or normalized weighted node flow to evaluate the load

balancing efficiency when employing the proposed algorithm for minimizing average controller load in SDWAN.

The rest of the paper is organized as follows: Section II introduces the MILP model formulation of SDWAN and formulates performance metrics mathematically with practical constraints. Section III introduces the proposed algorithm SAPKM and its variants by changing decision conditions from average load to weighted cost functions. Section IV presents the numerical results, which show the superiority of the low-complexity SAPKM in both minimizing average controller load and enhancing other performance metrics simultaneously. Finally, concluding remarks are drawn in Section V.

II. PROBLEM FORMULATION AND PERFORMANCE METRICS

In this section, the system model of the SDWAN is illustrated in the first place. We model the load balancing controller placement problem as a mixed-integer linear program (MILP) with performance metrics formulated mathematically. Two cost functions in perspective of topology structure and flow traffic distribution are defined to evaluate the appropriateness of the deployment.

A. PROBLEM FORMULATION

In this paper, we assume the SDWAN consists of multiple in-band controllers in the control plane and plenty of SDN forwarding devices, like OpenFlow switches and routers, in the data plane.

The load of the controllers is not only affected by the network topology structure but also limited by the processing capacity. Thereinto, the network topology may change in the following situations [31]. Firstly, if the flow traffic load surpasses the overall processing capacity of the distributed controllers, new controllers will be set up to prevent the network from paralysis. Secondly, if a controller is out of work due to a node or link failure, the switches in the corresponding sub-domain will connect to other controllers. Thirdly, if the load of an individual controller is beyond its processing capacity, selected switches will also be migrated to other controllers. If a controller is overloaded, both the response latency and the controller failure rate will increase. Thus, load balancing between the distributed controllers is critical for solving the controller placement problem. Due to the large coverage of the SDWAN, we assume that the network topology will remain static after the network is initialized.

The latency between switches and controllers is particularly critical since the connections between them are the fundamental basis for network management. Four kinds of latencies compose the overall network latency, i.e. packet transmission latency, packet propagation latency, switch queueing latency and controller processing latency [32]. For scenarios like WANs, e.g. OS3E, controller processing latency and packet propagation latency play significant roles in the overall latency; while for data centers, the rest two kinds of latencies dominate. Moreover, considering the dominance of propagation latency in transmission latency between controllers and switches, we mainly focus on the propagation latency here.

As for controller placements to attain reliable network, [9] defines reliability as a performance metric and tried to minimize the expected percentage of control path loss. In other words, controller placements can prevent the network from disruptions owing to node or link failures to a certain degree. Nevertheless, improper controller placements and assignments between switches and controllers may also cause links overload or load imbalance.

Therefore, this paper mainly considers the impact of controller placements on average controller load and other performance metrics, including average/maximum propagation latency and average network reliability. The optimization target is to minimize the average controller load in SDWAN with latency and processing capacity constraints. Besides, two cost functions are established to assess load balancing in terms of topology structure and flow traffic distribution. It should be noted that our main interest is to properly deploy a sufficient number of controllers to prevent the network from congestion or controller overload. In other words, issues like the optimum number of controllers and switch migrations due to overloaded controllers or network congestion are beyond the scope of this paper.

B. PLACEMENT METRICS

We assume an SDWAN is modeled as an undirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of nodes, and $\mathcal{E}$ the set of physical links among them. Detailed notations and definitions are summarized in Table 1.

For any switch $i \in \mathcal{V}$, d_{ij} is defined as the latency of the shortest path from switch i to controller j. According to previous works [1], the shortest path from i to j is selected by using the Dijkstra algorithm. Hence, the average and maximum latency can be defined respectively as

$$d_{ij} = \frac{Dijk(i,j)}{vel}, i \in \mathcal{V}, j \in \mathcal{C}, \tag{1}$$

$$D_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ij}d_{ij}, \tag{2}$$

$$D_{\max} = \max_{i \in \mathcal{V},} \sum_{j=1}^{N} x_{ij}d_{ij}. \tag{3}$$

The process of flow traffic distribution in each sub-domain can be described as an M/M/1 queue [33]. Thus, based on the M/M/1 queueing model in queuing theory, the controller in each sub-domain can be regarded as the server, while the switches can be regarded as the users served by the controller. In the proposed model of SDWAN, the controller load is defined as the flow request rate of all the switches in the corresponding sub-domain. Hence, the load of controller j

TABLE 1. Notations and Definitions.

Notation	Definition
$G(\mathcal{V}, \mathcal{E})$	a Physical network with node set $\mathcal{V}$ and link set $\mathcal{E}$
$\mathcal{V}$	the set of OpenFlow switch nodes
$\mathcal{C}$	the set of SDN controllers
M	the number of SDN controllers
N	the number of switch nodes
y_j	a controller is deployed on the position of switch j
x_{ij}	the assignment between switch i and a controller j
λ_i	the flow request rate of switch i
A_j	the processing capacity of controller j
η_j	redundancy factor of controller j
L_{avg}	average controller load
L_{dgr}	normalized weighted node degree
L_{flw}	normalized weighted node flow
d_{ij}	propagation latency from switch i to controller j
D_{avg}	average network propagation latency
D_{max}	maximum network propagation latency
D_{cst}	propagation latency constraint: maximum latency the network can tolerate
p_v	failure probability of a node v
p_e	failure probability of link e
r_{ij}	the path reliability from swithc i to controller j
R_{avg}	the average path reliability of all the shortest path from switches to corresponding controllers

and the average load can be expressed as following:

$$L_j = \frac{1}{n}\sum_{i=1}^{N} x_{ij}\lambda_{ij}, \tag{4}$$

$$L_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ij}y_j\lambda_{ij}. \tag{5}$$

To prevent the network from overload problem, we introduce two cost functions, L_{dgr} and L_{flw}, as load balancing indices to evaluate the load balancing issue, which represent the weighted node degree and weighted node flow, respectively. As for network topology structure, we not only consider the number of switches in each sub-domain but also think about the degrees of them. On the other hand, as for flow traffic distribution, we not only consider the flow rate of switches but also take the propagation latency between the controller and the switches into account in each sub-domain. Thus, we defined the two load balancing index of the model as following:

$$w_j = 1 + \frac{Deg_j}{Deg_{\text{max}}}, \tag{6}$$

$$\Phi_{j,\text{dgr}} = \sum_{i=1}^{N} x_{ij}w_j d_{ij}, \tag{7}$$

$$\Phi_{\text{dgr}} = \sum_{j=1}^{N} y_j \Phi_{j,\text{dgr}}, \tag{8}$$

$$L_{\text{dgr}} = \sqrt{\sum_{j=1}^{N} y_j\left(\frac{\Phi_{j,\text{dgr}}}{\Phi_{\text{dgr}}} - \frac{1}{M}\right)^2}, \tag{9}$$

$$\Phi_{j,\text{flw}} = \sum_{i=1}^{N} x_{ij}\lambda_{ij}d_{ij}, \tag{10}$$

$$\Phi_{\text{flw}} = \sum_{j=1}^{N} y_j \Phi_{j,\text{flw}}, \tag{11}$$

$$L_{\text{flw}} = \sqrt{\sum_{j=1}^{N} y_j\left(\frac{\Phi_{j,\text{flw}}}{\Phi_{\text{flw}}} - \frac{1}{M}\right)^2}. \tag{12}$$

Among them, eq. (6) means the weighted node degree for a node; eq. (7) means the weighted node degree for controller j; eq. (8) means the overall weighted node degree; eq. (9) means the normalized weighted node degree. Similarly, eq. (10) means the weighted node flow for controller j; eq. (11) means the overall weighted node flow; eq. (12) means the normalized weighted node flow.

Let $\mathcal{V}_{i\to j}$ denotes the node set, and $\mathcal{E}_{i\to j}$ denotes the link set on the path from switch i to controller j, the path reliability r_{ij} and the average network reliability R_{avg} can be calculated as

$$r_{ij} = \prod_{e\in\mathcal{E}_{i\to j}}(1-p_e)\prod_{v\in\mathcal{V}_{i\to j}}(1-p_v), \tag{13}$$

$$R_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{M}\sum_{j=1}^{N} x_{ij}y_j r_{ij}. \tag{14}$$

Therefore, the load balancing controller placement problem in SDWAN can be formally defined as follows.

Given a set of nodes $\mathcal{V}$, and M controllers for deployment, our goal is to determine the optimal placement of M controllers $\mathcal{C} \subseteq \mathcal{V}$ to minimize the average controller load as top priority, reduce the propagation latency, improve the network reliability, lower the cost function with latency and processing capacity constraints, namely,

$$\min \; L_{\text{avg}} \tag{15}$$

$$\text{s.t. } D_{\text{avg}} \le D_{\text{cst}}, \tag{16}$$

$$\sum_{i=1}^{N} x_{ij}\lambda_{ij} \le \eta_j A_j, \tag{17}$$

$$\sum_{j=1}^{N} x_{ij} = 1, \tag{18}$$

$$\sum_{j=1}^{N} y_j = M, \tag{19}$$

$$x_{ij} \le y_j, \tag{20}$$

$$x_{ij}, y_j \in \{0, 1\}, \quad \forall\, 1 \le i \le n,\; 1 \le j \le n \tag{21}$$

Among them, eq. (16) means that the average propagation latency should meet the latency constraints; eq. (17) means the flow traffic should meet the processing capacity constraints; eq. (18) means that each switch is assigned to merely one controller; eq. (19) means there are exactly M controllers deployed separately in the network; eq. (20) means there is no control path between node i and j without a controller placed at node j; eq. (21) are numeric constraints.

III. SIMULATED ANNEALING PARTITION-BASED K-MEANS ALGORITHM FOR LOAD BALANCING CONTROLLER PLACEMENT PROBLEM

By incorporating the network partition scheme, a hybridized efficient algorithm is proposed to address the load balancing controller placement problem in this section. For SDWAN, it is essential to partition the network for the deployment of distributed controllers. The core of network partition is to depart the whole networks into multiple sub-domains by using appropriate methods. Previous works [8], [9] apply simulated annealing (SA) algorithm and clustering-based network partition algorithm, i.e., Optimized K-Means (OKM) algorithm respectively to study the CPP and validate the effectiveness of the above algorithms for either enhancing reliability or decreasing end-to-end latency.

In the standard K-means algorithm, the initial nodes are randomly selected, and the network partition varies from each iteration. On the other hand, the main idea of OKM is to divide the topology into a certain number of sub-domains by iteratively partitioning the topology and updating the centroids of them. Meanwhile, the main thought of SA is to iterate a new neighbor solution every execution to continuously optimize the objective function, with random initialization.

However, previous works merely focused on optimizing the process of iteration but neglected the importance of node initialization. Thus, enlighted by the thought of OKM, we reconsider the problem of controller placements and switches assignments, proposing a Simulated Annealing Partition-based K-Means (SAPKM) algorithm to tackle the load balancing controller placemen problem with lower computational complexity.

Given topologies will be divided into multiple sub-domains to simplify the deployment process. The centroid is defined as the only node which has the shortest distance to other nodes in each sub-domain. Therefore the centroid initialization of a given network topology is unique and certain. It is worth mentioning that selecting the centroids as the initial sets contributes to further reducing the complexity of the problem.

It should be noted that the above algorithms are designed essentially in perspective of propagation latency, where the problems of "controller placement" and "switch assignment" are unified into the "controller placement problem." In other words, once the locations of the controllers are determined, the assignment relationship between the controllers and the switches are confirmed, then the network partition solutions are determined. Although the near-optimal or even optimal latency performance can be obtained, it is difficult to ensure that the proposed partition-based algorithm adaptiveness in load balancing when the difference of topology structures is considered. Therefore, it is necessary to make a difference between the "controller placement problem" and "switch assignment problem" for the load balancing controller placement problem towards SDWAN, where the optimization target is minimizing the controller load and maintaining the load balance in control plane.

The main idea of SAPKM can be divided into two steps: constantly partitions the given network topology from one sub-domain into a certain number of sub-domains based on the centroids to minimize the network propagation latency, and iteratively updates the centers based on the centroids of the sub-domains to minimize average controller load. Thereinto, the ensuing partition process of SAPKM in the first phase can ultimately find the determined centroids of the sub-domains, which guarantees both the less average controller load and less propagation latency within less computational time. Besides, the placement of the controllers obtained by executing SAPKM can also enhance the performance of network reliability to some extent.

As shown in Algorithm 1, the first step is to select initial sets, $\mathcal{C}_{\text{ctd}}$, which consist the centroids of M sub-domains. Then, we calculate the performance metrics and cost functions, including average controller load L_{avg}, average propagation latency D_{avg}, average network reliability R_{avg}, normalized weighted node degree L_{dgr}, and normalized weighted node flow L_{flr}. After entering the while loop, a new neighbor solution $\mathcal{C}_{\text{new}}$ will be obtained at every iteration. If the updated average network latency D_{avg} is lower than the threshold latency D_{cst} and the each updated controller load is less than corresponding available processing capacity, the updated network reliability R_{new} and two cost functions, $L_{\text{dgr\_new}}, L_{\text{flw\_new}}$, will be calculated and recorded. In the process of minimizing average controller load, some unsatisfied results may also be recorded and updated with the acceptance probability $P(\Delta) = e^{-\frac{\Delta}{T}}$, where

$$\Delta = L_{\text{new}} - L_{\text{avg}}. \tag{22}$$

In addition, $T = T_0$ denotes the initial temperature, T_{final} the terminate termperature, and α the annealing coefficient. It should be noted that, the difference value can also be defined by the cost function L_{dgr} or L_{flw}, namely:

$$\Delta = L_{\text{dgr\_new}} - L_{\text{dgr}}, \tag{23}$$

$$\Delta = L_{\text{flr\_new}} - L_{\text{flr}}. \tag{24}$$

Finally, we obtain the optimized placement results with minimized average controller load and other enhanced performance metrics under practical constraints.

IV. SIMULATION RESULTS

The proposed SAPKM formulation is evaluated on various real networks from OS3E [34] and Internet Topology Zoo [35] with the comparisons on the performance metrics with other algorithms. Specifically, five network topologies with various structures and number of nodes are included: Nsfnet, ATT, Agis, OS3E, Chinanet, with detailed topology and failure probability settings listed in Table 2 [36]. In order to obtain stable performance, we repeat random placement algorithms for 100 times and take the average value. We assume the flows transmitted by the switches meet the Poisson distribution with the flow request rate ranges from 150 kb/s to 550 kb/s. Besides, the processing capacity of each

Algorithm 1 Simulated Annealing Partition-based K-Means algorithm (SAPKM) for Load Balancing Controller Placement

```
Input:
    $G(\mathcal{V}, \mathcal{E})$: the target topology of a certain physical network.
    $M$: the target number of SDN controllers.
    $\boldsymbol{A}$: the processing capacity sequence of controllers.
    $\boldsymbol{\eta}$: the redundancy factor sequence of controllers.
    $\boldsymbol{\lambda}$: the flow request rate sequence of switches.
    $D_{\rm cst}$: the network latency constraint.
Output:
    $\mathcal{C}_{\rm opt}$: the optimized placement of $M$ SDN controllers for load balancing.
    $D_{\rm avg}$: the average propagation latency between controllers and switches.
    $R_{\rm avg}$: the average path reliability of the netwotk.
    $\boldsymbol{L}$: the load sequence of controllers.
    $L_{\rm dgr}$: the cost of normalized weighted controller degree.
    $L_{\rm flw}$: the cost of normalized weighted controller flow.
 1: Initialize $T = T_0, T_{\rm final}, \alpha$.
 2: Partition the topology into $M$ sub-domains, obtain $M-centroids$ set $\mathcal{C}_{\rm ctd}$.
 3: Compute $D_{\rm avg}, R_{\rm avg}, \boldsymbol{L}, L_{\rm dgr}, L_{\rm flw}$ according to $\mathcal{C}_{\rm ctd}$.
 4: while $T > T_{\rm final}$ do
 5:     Generate a new neighbor controller set $\mathcal{C}_{\rm new}$.
 6:     Compute $D_{\rm new}, \boldsymbol{L}_{\rm new}$, according to $\mathcal{C}_{\rm new}$.
 7:     if $D_{\rm new} \le D_{\rm cst}, \boldsymbol{L}_{\rm new} \le \boldsymbol{\eta A}$ then
 8:         Compute $R_{\rm new}, L_{\rm dgr\_new}, L_{\rm flw\_new}$.
 9:         $\Delta = \boldsymbol{L}_{\rm new} - \boldsymbol{L}$.
10:         Generate a random number $\delta \in (0, 1)$.
11:         if $\Delta \le 0$ or $e^{-\Delta/T} > \delta$ then
12:             $\mathcal{C}_{\rm opt} = \mathcal{C}_{\rm new}$.
13:             $D_{\rm avg} = D_{\rm new}, R_{\rm avg} = R_{\rm new}, \boldsymbol{L} = \boldsymbol{L}_{\rm new}$,
                $L_{\rm dgr} = L_{\rm dgr\_new}, L_{\rm flw} = L_{\rm dgr\_new}$.
14:         end if
15:     end if
16:     $T = T \cdot \alpha$.
17: end while
18: return $\mathcal{C}_{\rm opt}, D_{\rm avg}, R_{\rm avg}, \boldsymbol{L}, L_{\rm dgr}, L_{\rm flw}$
```

TABLE 2. Topology and Failure Probability Settings.

Topology	Number of nodes	Number of links	P_v	P_e
Nsfnet	13	15	[0, 0.05]	[0, 0.02]
ATT	25	57	[0, 0.06]	[0, 0.04]
Agis	25	30	[0, 0.06]	[0, 0.04]
OS3E	34	41	[0, 0.07]	[0, 0.06]
Chinanet	38	62	[0, 0.08]	[0, 0.08]

controller is set to 15M with the redundancy factor randomly selected between 0.9 and 1 [29].

For better estimating the load balancing efficiency of the above algorithms, we record the average deployment results by two cost function in terms of the weighted node degree and the weighted node flow, respectively. All the algorithms are performed in MATLAB (R2018a) running on a MacBook Air, with 1.8GHz Intel Core i5 CPUs, 8GB 1600MHz DDR3 RAM.

The simulation consists of four following steps: First of all, topology settings including node coordinates and the links connections are derived from real network topologies on the public. With the length of each available path calculated by the Haversine formula [4], the Dijkstra algorithm is employed to choose the shortest path between any two nodes. Furthermore, the associated propagation latency is calculated with propagation velocity, $vel = 2 \times 10^8\ m/s$ [32]. Finally, with the above preliminary works and topology settings on failure probability, various algorithms are employed to solve the CPP for minimizing average controller load and enhance other performance metrics under latency and processing capacity constraints. Here, we assume the constraint latency is 10ms in the following simulations.

In order to evaluate the performance and computational complexity of the proposed algorithm, we compare SAPKM with other two representative solutions, SA, and OKM. The effectivenesses of both SA and OKM on CPP are exhaustively proved in previous works [8], [9].

We first evaluate the SAPKM algorithm in comparison with the above solutions and then demonstrate how the average controller load is minimized and cost functions are decreased in large topology with less running time. Throughout this paper, we use cost function and load balancing index interchangeably.

A. LOAD DISTRIBUTION AND LOAD BALANCING

In this subsection, we first analyze two cost functions of the above network topologies when employing various algorithms. In order to give insight into the cost functions of the controller load, we take the OS3E topology as an example to analyze the effect of the number of controllers on them. Besides, for exploring the relationship between the two cost functions, we make a comparison in the values of both of them. Specifically, we employ the proposed algorithm by changing the decision conditions in turn. For example. if we are interested in the load balancing efficiency in terms of the average weighted node degree, we can employ the proposed algorithm twice by using both cost functions as the decision conditions, respectively. In the experiments, we simultaneously record the load balancing index values of both weighted node degree and weighted node flow. Finally, we compare and analyze the relationship between the flow traffic distribution and the cost functions in detail.

Fig. 1 shows the average weighted node degree load balancing index of three algorithms in different network topologies when $M = 5$. The weighted node degree cost function represents the number of switches managed by each controller in the corresponding sub-domain. Besides, we also consider the node degree of each switch and the propagation latency from each of them to the controller. In perspective of the network topology, the smaller the average weighted node

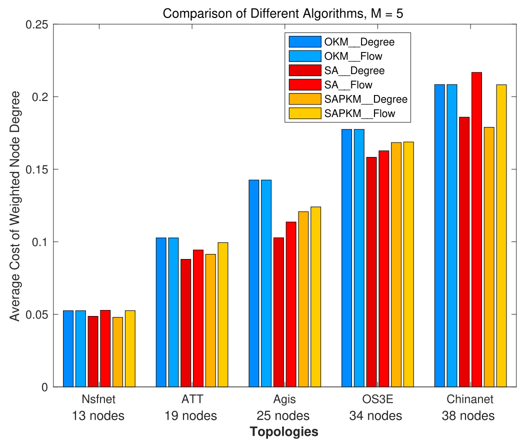

FIGURE 1. Comparisons of the average weighted node degree on different network topologies.

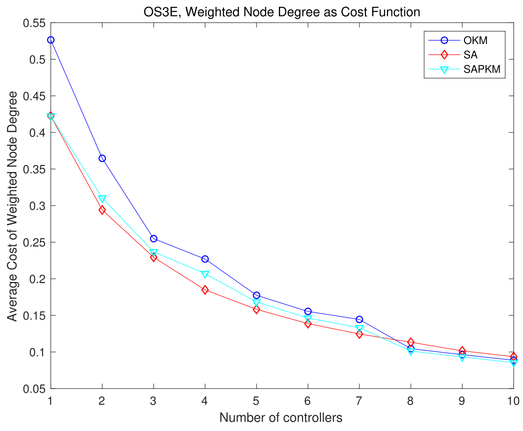

FIGURE 2. Comparisons of the average weighted node degree on OS3E.

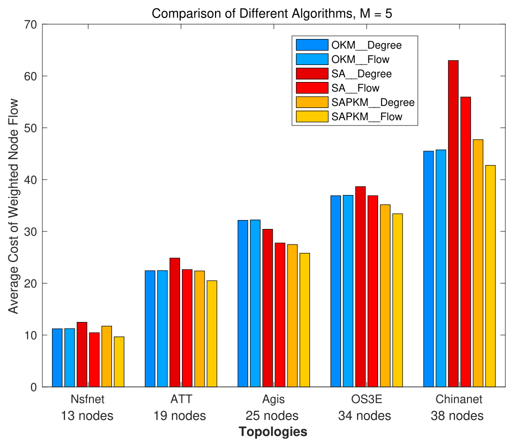

FIGURE 3. Comparisons of the average weighted node flow on different network topologies.

degree cost function, the more balanced the controller load in each sub-domain. Apparently, the more nodes in the network topology, the larger the values of the average weighted node degree cost function. As a matter of course, the values of average weight node degree derived from three algorithms in weighted node degree have better efficiency of load balancing than that of weighted node flow. In other words, the weighted node flow as the cost function cannot achieve better values for the weighted node degree.

Besides, both SA and SAPKM almost have less value of average weighted node degree than OKM. However, due to the different topological properties of the above networks, SAPKM outperforms SA in Nsfnet and Chinanet, while the opposite conclusions are drawn in ATT, Agis, and OS3E.

Therefore, if we defined the weighted node degree cost function as the load balancing index, the proposed algorithm may achieve good performance, but the load balancing efficiency may be limited by the topological properties.

Fig. 2 shows the average weighted node degree cost function of three algorithms in the OS3E topology when the number of controllers varies from 1 to 10. Conspicuously, the average weighted node degree cost function has a monotone decreasing tendency with a growing number of controllers. SA has better load balancing efficiency than SAPKM and OKM when the number of controllers is less than or equal to 7. On the other hand, SAPKM and OKM have smaller values when 8 or more controllers are deployed in the OS3E topology, and SAPKM outperforms PKM on this occasion. Therefore, SAPKM has near-optimal load balancing index in weighted node degree, but the efficiency of load balancing still subject to the topological properties.

Similarly, Fig. 3 shows the average weighted node flow load balancing index of the above algorithms in different network topologies when $M = 5$. The weighted node flow cost function represents the aggregated number of flow requests from the switches managed by the controller per second in the corresponding sub-domain. Besides, we also consider the propagation latency from each switch to the controller. In perspective of the flow traffic distribution, the smaller the average weighted node flow cost function, the more balanced the controller load in each sub-domain. Apparently, the more nodes in the network topology, the larger the values of the average weighted node flow cost function. As a matter of course, the values of average weight node flow derived from three algorithms in weighted node flow have better efficiency of load balancing than that of weighted node degree. In other words, the weighted node degree as the cost function cannot achieve better values for the weighted node flow. In contrast, SAPKM always has less value of average weighted node flow than both OKM and SA. Furthermore, even the above networks have various topological properties, SAPKM with weighted node flow as the load balancing index achieves the best load balancing efficiency than other algorithms no matter the load balancing index is set to weighted node degree or weighted node flow.

Therefore, if we defined the weighted node flow cost function as the load balancing index, the proposed algorithm can achieve near-optimal load balancing performance, which is not limited by the topological properties.

Fig. 4 shows the average weighted node flow load cost function of three algorithms in the OS3E topology when the number of controllers varies from 1 to 10. Conspicuously, the average weighted node flow cost function has a monotone

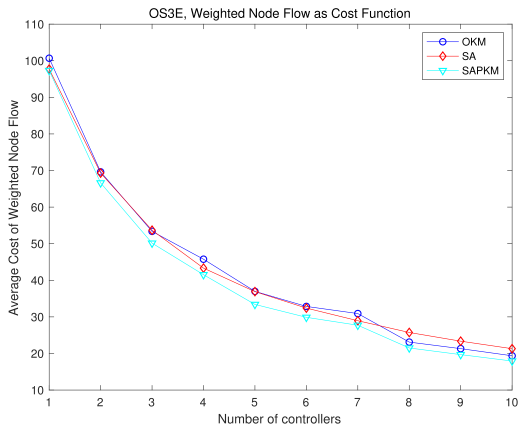

FIGURE 4. Comparisons of the average weighted node flow on OS3E.

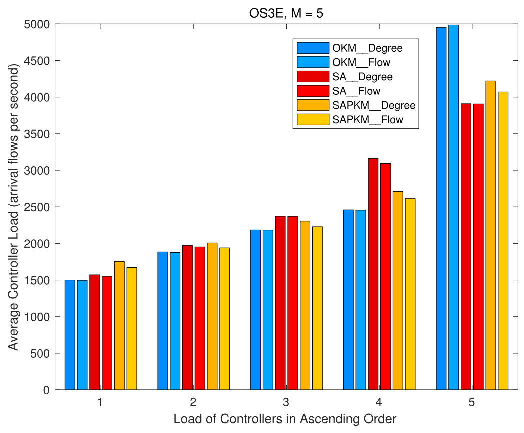

FIGURE 6. Comparisons of average load sequence on OS3E in normalized cost functions.

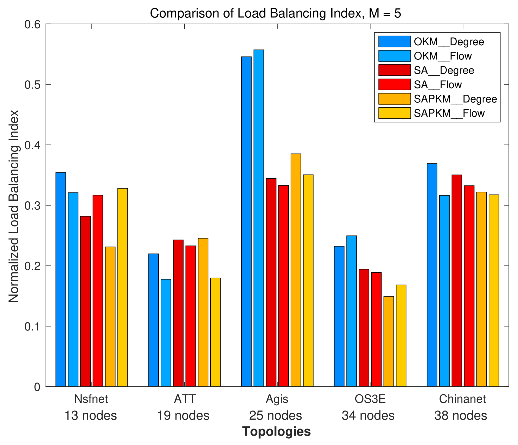

FIGURE 5. Comparisons of the normalized load balancing index on different network topologies.

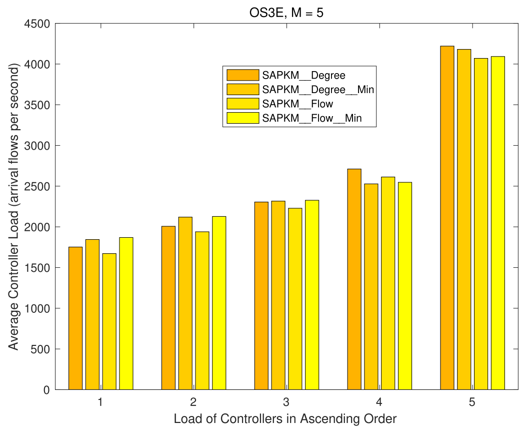

FIGURE 7. Comparisons of average load sequence on OS3E with SAPKM in minimum/average normalized cost functions.

decreasing tendency with a growing number of controllers. Compared with the weighted node degree cost function, SAPKM with weighted node flow load balancing index always has better load balancing efficiency than SA and OKM. In addition, it should be noted that OKM has a smaller value than SA does when 8 or more controllers are deployed in the OS3E topology. Therefore, SAPKM has near-optimal load balancing index in weighted node degree, and the efficiency of load balancing is not subject to the topological properties.

In order to further explore the rationality of the above cost functions, we perform the algorithms in the above network topologies and analyze the two normalized cost functions, L_{dgr} and L_{flw}, instead of average cost functions as stated above. We also record the average controller load sequence in ascending order to show the efficiency of the proposed algorithm with different cost functions.

Fig. 5 shows the normalized load balancing index of the above algorithms with different cost functions in the above network topologies when $M = 5$. The normalized load balancing index represents the standard deviation of the cost function among the controllers. Unexpectedly, the value of the normalized load balancing index varies from each other in the above topologies. In general, the proposed algorithm always has better load balancing efficiency in the above topologies except for Nsfnet. Since the normalized load balancing index mainly evaluate how much the weight cost function of the controllers is from the average value, the comparison in normalized indices weakens the effect of topological properties and actual flow traffic distribution. Thus how to evaluate the load balancing efficiency by taking both topological properties and actual flow traffic distribution into account worth further study, which is one of our future research directions. Hence we take the OS3E topology as an example to further study the relationship between the various load balancing index and the controller load sequence in terms of flow traffic distribution.

Fig. 6 shows the average controller load sequence in ascending order in the OS3E topology by employing three algorithms with two cost functions. It should be noted that, the average controller load sequence in Fig. 6 is derived from repeating the algorithms and taking the average value. Meanwhile, we further analyze the effect of two cost functions on the controller load sequence by performing the proposed algorithm.

Fig. 7 shows the average controller load sequence in ascending order in the OS3E topology by employing SAPKM

TABLE 3. Normalized Load Balancing Index in OS3E by Performing SAPKM.

	Normalized Load Balancing Index
Average Normalized Weighted Node Degree	0.1490
Minimum Normalized Weighted Node Degree	0.1030
Average Normalized Weighted Node Flow	0.1682
Minimum Normalized Weighted Node Flow	0.1101

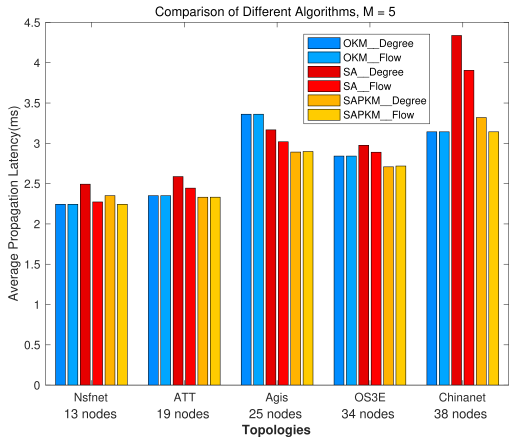

FIGURE 8. Comparisons of the overall average propagation latency on different topologies.

with two cost functions. We not only display the average results of SAPKM by applying two cost functions but also record the best results of the two functions for comparison. Apparently, the load distribution is similar no matter what the cost function is taken when employing SAPKM regardless of the cost function. Table 3 illustrated the normalized load balancing indices corresponding to the controller load sequence in Fig. 7.

Therefore, we can deduce that the proposed algorithm SAPKM not only decreases the average controller load but also enhance the efficiency of load balancing in lowering the cost and load distribution.

B. PROPAGATION LATENCY

In this subsection, we first compare the average propagation latency on the above topologies with three algorithms, then take the OS3E topology as an example to analyze the effect of the number of controllers on average propagation latency.

Fig. 8 displays the comparison results of average propagation latency on various topologies by employing the above algorithms when $M = 5$. For OKM, both cost functions show almost the same average propagation latency performance. For SA, taking weighted node flow as the cost function shows better latency performance than weighted node degree cost function does. On the other hand, SA outperforms OKM for latency performance in Agis, while OKM surpasses SA in other network topologies. For SAPKM, taking weighted node

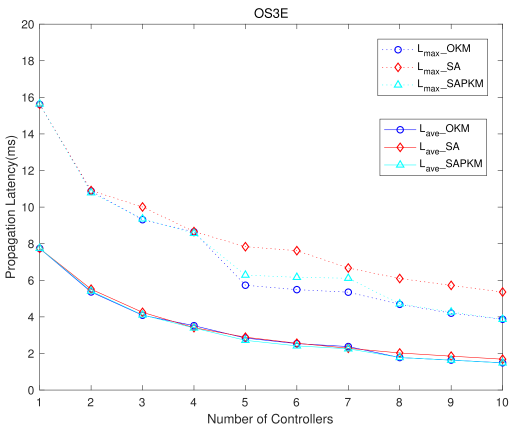

FIGURE 9. Comparisons of the overall average propagation latency on OS3E with different number of controllers.

flow as the cost function shows the best average propagation latency in the above topologies in comparison with other algorithms. Besides, if we defined the weighted node degree cost function as the load balancing index, latency performance will not be optimal in Nsfnet and Chinanet topologies for SAPKM. Thus, we further investigate the effect of the number of controllers on the average propagation latency by taking the weighted node flow as cost function.

C. NETWORK RELIABILITY

Similarly, we first compare the average network reliability on the above topologies with three algorithms, then take the OS3E topology as an example to analyze the effect of the number of controllers on reliability performance.

Fig. 9 shows the comparison results of average/maximum propagation latency on topology OS3E among the above algorithms when the number of controllers varies from 1 to 10. Apparently, the propagation latency presents a monotone decreasing tendency as the number of controllers increases. Besides, all the algorithms show similar average propagation latency performance with the increasing number of controllers, where SAPKM slightly outperform the other two algorithms. However, OKM has the best maximum propagation latency than SA and SAPKM. Thereinto, SAPKM has almost the same maximum propagation latency except $M \in \{5, 6, 7\}$.

Therefore, SAPKM with weighted node flow as the cost function achieves near-optimal average/maximum propagation latency performance for solving the load balancing issue in SDWAN.

Fig. 10 demonstrates the comparison results of average network reliability on various topologies by employing the above algorithms when $M = 5$. For OKM, both cost functions show almost the same average network reliability performance and outperform SA and SAPKM except OS3E. For SA, similarly, taking weighted node flow as the cost function shows better reliability performance than weighted node degree cost function does. However, SA cannot guarantee reliability performance in comparison with OKM and SAPKM. Obviously,

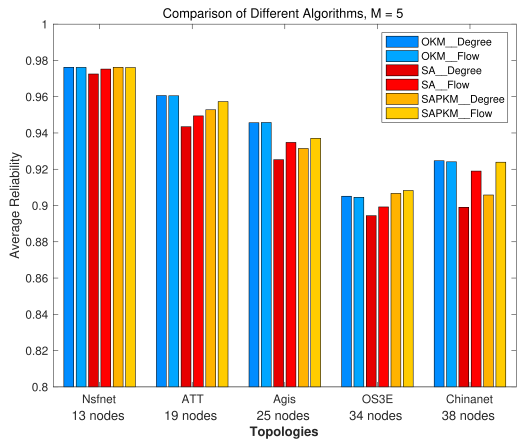

FIGURE 10. **Comparisons of the overall average reliability on different topologies.**

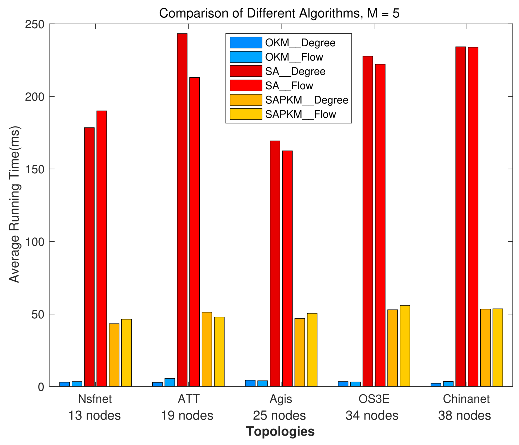

FIGURE 12. **Comparisons of the overall average running time on different topologies.**

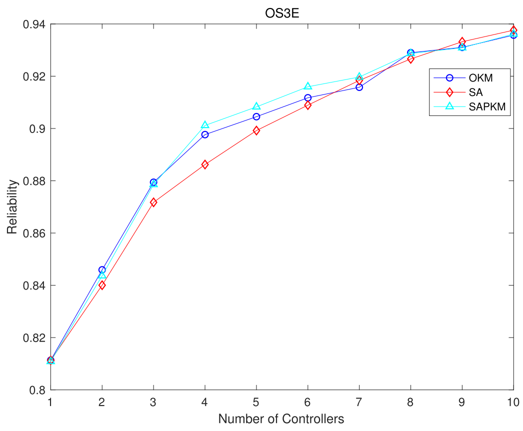

FIGURE 11. **Comparisons of the overall average reliability on OS3E with different number of controllers.**

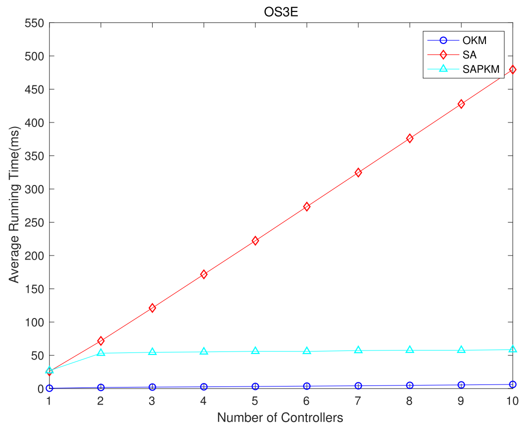

FIGURE 13. **Comparisons of the overall average running time on OS3E with different number of controllers.**

SAPKM shows the same tendency for the superiority of taking weighted node flow as the cost function in enhancing reliability. In addition, taking weighted node flow as the cost function shows the near-optimal reliability performance compared with OKM in the above topologies, while SAPKM surpasses OKM on topology OS3E. Thus, we further investigate the effect of the number of controllers on the average network reliability by taking the weighted node flow as cost function.

Fig. 11 shows the comparison results of average network reliability on topology OS3E among the above algorithms when the number of controllers varies from 1 to 10. Apparently, the network reliability presents a monotone increasing tendency as the number of controllers increases. Both SAPKM and OKM outperform SA when less than 8 controllers are deployed in OS3E. It should be noted that SAA slightly exceeds OKM when the number of controllers is 7. Besides, SAA ultimately surpasses OKM and SAPKM when deploying 9 or 10 controllers.

Therefore, SAPKM with weighted node flow as the cost function achieves near-optimal average network reliability for solving load balancing issue in SDWAN when 8 or fewer controllers are deployed on OS3E.

D. COMPUTATIONAL COMPLEXITY

Fig. 12 depicts comparisons of the computational running time on the above topologies by employing the above algorithms when $M = 5$. Apparently, the computational complexities of both OKM and SAPKM are more efficient than that of SA. SAPKM needs almost 50ms to converge while OKM needs 10ms at most for convergence, which is suitable for online use. Owing to the efficient and available initialization by selecting the centroids as the controllers, plenty of redundant iterations in SA will not take place in SAPKM. Similarly, we further investigate the effect of the number of controllers on the computational running time by taking the weighted node flow as cost function.

Fig. 13 shows the comparison results of computational running time on topology OS3E among the above Obviously, the acceleration of the number of iterations with an increasing amount of controllers maintains stable when employing SAPKM, while the running time of SA improves stably as the number of controllers increases.

Therefore, compared to SA, SAPKM greatly shortens the computational time to ensure minimizing average controller load with enhancing latency and reliability performance.

Although SAPKM consumes more computational resources than OKM, it is still advantageous to online cases.

V. CONCLUSION

In this paper, we investigated the load balancing problem in SDWAN and developed a low-complexity algorithm SAPKM to minimize average controller load. We formulated the load balancing controller placement problem as a MILP problem in order to minimize average controller load and other performance metrics, including propagation latency and network reliability, under latency and processing capacity constraints. Two cost functions in terms of topology structure and flow traffic were also defined to evaluate the appropriateness of the deployment on load balancing efficiency towards SDWAN. Numerical results proved the effectiveness of SAPKM in both load balancing and other performance metrics for taking both the topology structure and flow traffic distribution into consideration. Furthermore, taking the weighted node flow as the cost functions in the proposed algorithm achieves near-optimal performance in both average controller load and load distribution.

We will further study the load balancing problem for topologies with multi-degree internal nodes density in the future. Besides, exploring new efficient algorithms towards multi-objective optimization in SDWAN is also part of our future works.

REFERENCES

[1] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proc. ACM HotSDN*, 2012, pp. 7–12.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.

[3] B. Pfaff. *OpenFlow Switch Specification 1.3.0*. Accessed: Oct. 15, 2018. [Online]. Available: https://3vf60mmveq1g8vzn48q2o71a-wpengine.netdna-ssl.com/wp-content/uploads/2014/10/open_ow-spec-v1.3.0.pdf

[4] P. Xiao, W. Qu, H. Qi, Z. Li, and Y. Xu, "The SDN controller placement problem for WAN," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Oct. 2014, pp. 220–224.

[5] A. Sallahi and M. St-Hilaire, "Optimal model for the controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 19, no. 1, pp. 30–33, Jan. 2015.

[6] 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 Trans. Netw. Service Manag.*, vol. 12, no. 1, pp. 4–17, Mar. 2015.

[7] T. Zhang, A. Bianco, and P. Giaccone, "The role of inter-controller traffic in SDN controllers placement," in *Proc. NFV-SDN*, Nov. 2016, pp. 87–92.

[8] G. Wang, Y. Zhao, J. Huang, and Y. Wu, "An effective approach to controller placement in software defined wide area networks," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 1, pp. 344–355, Mar. 2018.

[9] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, "On reliability-optimized controller placement for software-defined networks," *China Commun.*, vol. 11, no. 2, pp. 38–54, Feb. 2014.

[10] L. F. Müller, R. R. Oliveira, M. C. Luizelli, L. P. Gaspary, and M. P. Barcellos, "Survivor: An enhanced controller placement strategy for improving SDN survivability," in *Proc. IEEE GLOBECOM*, Dec. 2014, pp. 1909–1915.

[11] F. J. Ros and P. M. Ruiz, "Five nines of southbound reliability in software-defined networks," in *Proc. ACM Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 31–36.

[12] Y. Jiménez, C. Cervelló-Pastor, and A. J. García, "On the controller placement for designing a distributed SDN control layer," in *Proc. IFIP Netw. Conf.*, 2014, pp. 1–9.

[13] 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 *Proc. IEEE CNSM*, Oct. 2013, pp. 18–25.

[14] M. T. I. ul Huque, W. Si, G. Jourjon, and V. Gramoli, "Large-scale dynamic controller placement," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 1, pp. 63–76, Mar. 2017.

[15] K. Guo, M. Lin, B. Zhang, W.-P. Zhu, J.-B. Wang, and T. A. Tsiftsis, "On the performance of LMS communication with hardware impairments and interference," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1490–1505, Feb. 2019.

[16] K. Guo, K. An, B. Zhang, Y. Huang, and D. Guo, "Physical layer security for hybrid satellite terrestrial relay networks with joint relay selection and user scheduling," *IEEE Access*, vol. 6, pp. 55815–55827, 2018.

[17] K. Guo, K. An, Y. Huang, and B. Zhang, "Physical layer security of multiuser satellite communication systems with channel estimation error and multiple eavesdroppers," *IEEE Access*, vol. 7, pp. 96253–96262, 2019.

[18] K. Guo, K. An, B. Zhang, Y. Huang, D. Guo, G. Zheng, and S. Chatzinotas, "On the performance of the uplink satellite multiterrestrial relay networks with hardware impairments and interference," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2297–2308, Sep. 2019.

[19] B. Li, Z. Fei, X. Xu, and Z. Chu, "Resource allocations for secure cognitive satellite-terrestrial networks," *IEEE Wireless Commun. Lett*, vol. 7, no. 1, pp. 78–81, Feb. 2018.

[20] Y. Ai, A. Mathur, M. Cheffena, M. R. Bhatnagar, and H. Lei, "Physical layer security of hybrid satellite-FSO cooperative systems," *IEEE Photon. J.*, vol. 11, no. 1, Feb. 2019, Art. no. 7900814.

[21] B. Li, Z. Fei, Z. Chu, F. Zhou, K.-K. Wong, and P. Xiao, "Robust chance-constrained secure transmission for cognitive satellite–terrestrial networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4208–4219, May 2018.

[22] M. Jia, X. Gu, Q. Guo, W. Xiang, and N. Zhang, "Broadband hybrid satellite-terrestrial communication systems based on cognitive radio toward 5G," *IEEE Wireless Commun.*, vol. 23, no. 6, pp. 96–106, Dec. 2016.

[23] A. Papa, T. De Cola, P. Vizarreta, M. He, C. M. Machuca, and W. Kellerer, "Dynamic SDN controller placement in a LEO constellation satellite network," in *Proc. IEEE GLOBECOM*, Dec. 2018, pp. 1–6.

[24] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in *Proc. ACM SIGCOMM HoTSDN*, Aug. 2013, pp. 7–12.

[25] R. Ahmed and R. Boutaba, "Design considerations for managing wide area software defined networks," *IEEE Commun. Mag.*, vol. 52, no. 7, pp. 116–123, Jul. 2014.

[26] L. Yao, P. Hong, W. Zhang, J. Li, and D. Ni, "Controller placement and flow based dynamic management problem towards SDN," in *Proc. IEEE Int. Conf. Commun. Workshop (ICCW)*, Jun. 2015, pp. 1–6.

[27] A. Sallahi and M. St-Hilaire, "Expansion model for the controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 274–277, Feb. 2017.

[28] Y. Tingting, H. Xiaohong, M. Maode, and Y. Jie, "Balance-based SDN controller placement and assignment with minimum weight matching," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[29] G. Li, X. Wang, and Z. Zhang, "SDN-based load balancing scheme for multi-controller deployment," *IEEE Access*, vol. 7, pp. 39612–39622, 2019.

[30] X. Hou, W. Muqing, L. Bo, and L. Yifeng, "Multi-controller deployment algorithm in hierarchical architecture for SDWAN," *IEEE Access*, vol. 7, pp. 65839–65851, 2019.

[31] C. Wang, B. Hu, S. Chen, D. Li, and B. Liu, "A switch migration-based decision-making scheme for balancing load in SDN," *IEEE Access*, vol. 5, pp. 4537–4544, 2017.

[32] G. Wang, Y. Zhao, J. Huang, and W. Wang, "The controller placement problem in software defined networking: A survey," *IEEE Netw.*, vol. 31, no. 5, pp. 21–27, Sep./Oct. 2017.

[33] K. Atefi, S. Yahya, A. Rezaei, and A. Erfanian, "Traffic behavior of local area network based on M/M/1 queuing model using poisson and exponential distribution," in *Proc. IEEE Region 10 Symp. (TENSYMP)*, pp. 19–23, May 2016.

[34] *Internet2 Open Science, Scholarship and Services Exchange*. Accessed: Oct. 1, 2010. [Online]. Available: http://www.internet2.edu/network/ose/

[35] *The Internet Topology Zoo*. Accessed: Jun. 4, 2019. [Online]. Available: http://www.topology-zoo.org/dataset.html

[36] J. Liu, Y. Shi, L. Zhao, Y. Cao, W. Sun, and N. Kato, "Joint placement of controllers and gateways in SDN-enabled 5G-satellite integrated network," *IEEE J. Sel. Area Commun.*, vol. 36, no. 2, pp. 221–232, Feb. 2018.

KONGZHE YANG received the B.S. and M.S. degrees from the PLA University of Science and Technology, Nanjing, China, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree with the College of Communications Engineering, Army Engineering University of PLA, Nanjing. His research interests focus on anti-jamming communications, channel coding, software-defined networking (SDN), and satellite communications.

DAOXING GUO received the B.S., M.S., and Ph.D. degrees from the Institute of Communications Engineering (ICE), Nanjing, China, in 1995, 1999, and 2002, respectively. He is currently a Full Professor and also a Ph.D. Supervisor with the Army Engineering University of PLA. He has authored and coauthored more than 40 conference and journal articles and has been granted over 20 patents in his research areas. He has served as a reviewer for several journals in communication field. His current research interests include satellite communications systems and transmission technologies, communication anti-jamming technologies, and communication anti-interception technologies, including physical layer security and so on.

BANGNING ZHANG received the B.S. and M.S. degrees from the Institute of Communications Engineering (ICE), Nanjing, China, in 1984 and 1987, respectively. He is currently a Full Professor and the Head of the College of Communications Engineering, Army Engineering University of PLA. He has authored and coauthored more than 80 conference and journal articles and has been granted over 20 patents in his research areas. He has served as a reviewer for several journals in communication field. His current research interests include communication anti-jamming technologies, microwave technologies, satellite communications systems, cooperative communications and physical layer security.

BING ZHAO received the B.S. and M.S. degrees from the Nanjing University of Science and Technology (NJUST), Nanjing, China, in 2007 and 2009, respectively. She is currently a Full Lecturer with the Army Engineering University of PLA. She has been granted over ten patents in her research areas. Her current research interests include satellite communication systems and transmission technologies.