The Journal of Supercomputing
https://doi.org/10.1007/s11227-022-04360-3

Star capacity-aware latency-based next controller placement problem with considering single controller failure in software-defined wide-area networks

Hadi Mojez[1] · Amir Massoud Bidgoli[1] · Hamid Haj Seyyed Javadi[2]

Accepted: 5 February 2022

Abstract

The failure of a single controller in the control layer of the Software-Defined Network (SDN) challenges the management of switches in the data layer and disrupts the proper function of the network; therefore, the optimal placement of multiple controllers in the network and the assignment of switches to controllers, called Controller Placement Problem (CPP), is an emerging and very important issue. Because, in case of controller failures, switches must be reassigned to active controllers in the network, this process results in a significant increase in the worst-case latency after reassignment due to lack of planning for controller failures. For this reason, the problem can be customized for the network real world by planning the star assignment of switches to controllers despite failures. Also, for placing controllers, another challenge is the network search space, which can be significantly reduced by using standard array decision variables. In this paper, a new star capacity-aware delay-based controller placement approach with single controller failure is formulated and presented as a Mixed Integer Program (MIP). The purpose of the proposed approach is to minimize the maximum, for all switches, of the sum of the worst-case latency from the switch to the nearest first controller with enough capacity and the worst-case latency from the same switch to the closest second controller with enough capacity. We also use the population-based simulated annealing algorithm for fast convergence of the problem toward the optimal solution on large-scale networks. The proposed formulation and algorithm are estimated with the real Internet Topology Zoo and the results were extensively analyzed. Unlike controller placement approaches with linear (hierarchy) strategy, our proposed approach which is based on star planning and uses the array switch-to-controller assignment variables. According to the simulation results, our proposed approach performs better than Capacitated Next Controller Placement (CNCP) and Resilient Capacity-aware Controller Placement Problem (RCCPP) approaches in case of controller failure and in failure free case.

Extended author information available on the last page of the article

Published online: 16 March 2022

Keyword Star Planning · Software-defined wide-area network · Latency · Population-based simulated annealing algorithm

1 Introduction

A Software-Defined Network (SDN) is designed as an emerging structure with the abstraction and separation of different separable planes. It transfers the control layer functions of network elements such as routing and load distribution from within network elements to one or more logically and physically distributed centralized entities called controllers. Also, it retains the functions such as packet forwarding, policing, and queuing within network elements to simplify and improve management of the network with high resilience; therefore, network programming capability is enhanced through open interfaces, which leads to more innovative opportunities in these networks. In addition to the above objectives, SDNs can be combined with network functions virtualization technology for efficient network approach in terms of hardware costs [1, 2]. Also, SD-WAN technology [3] is a unique approach for software-defined wide-area networks because it combines traditional WAN technologies such as MPLS and broadband connections. This allows organizations to direct traffic to remote locations and different branches, while reducing network costs. Therefore, one controller is not enough to manage SDN. Because of this challenge, each forwarding device needs to be assigned to more than one controller (one controller as the master controller and the rest of the controllers as the backup controllers) to ensure reliability. In other words, it is necessary to deploy multiple controllers in locations that lead to optimal performance. All controllers must sustain a stable global view during placement in the network to ensure convenient network function. In fact, by placing multiple controllers in the SDN, we lead to a resilient and reliable design of the control plane. In other words, the Controller Placement Problem (CPP), first invented by Heller et al. [4], refers to determination of the number and location of controllers and assignment of forwarding devices to controllers in the network. CPPs that take into account both the capacity and reliability parameters of the controller are called capacity-aware reliable CPPs. In many previous researches, both the capacity and reliability parameters of the controller were considered and the objective function was to minimize latency. The challenges under consideration are the large network search space and the linear k-center problem not being customized for the network real world.

Killi and Rao [5] have proposed two mathematical optimization models for the cost-based resilient capacitated CPP by using constraints on the delay between switches and controllers. The purpose is to create high resilience against controller failures, reduce controller installation cost, and reduce the reserved backup extra capacity of controllers assuming the number of controllers is constant. However, they did not take the latency into account in their objective function, and in the formulation provided by them, the decision variable M_{ij} was used to assign the switches to the controllers, in which network search space, assuming N switches and C controllers in network, was equivalent to NC bit; but this is a relatively large search

space for deploying controllers in the network. In [2], Tanha et al. examined the Resilient Capacitated Controller Placement Problem (RCCPP) to minimize the number of controllers in order to have a cost-effective design, considering only the failure of one or more controllers. Also, similar to [5], they used the decision variable X_{ij} for the assignment operation, which results in a large search space in the network. Also, their approach, according to [5], is not suitable for delay-sensitive applications because the proposed formulation is based on the k-center or k-medium facility location problem, which is more focused on cost. In approaches of [2, 5], the performance metric is not the linear k-center but the star k-center, but the network search space is not optimal depending on the non-standard decision variables used. In [6], Killi and Rao studied reliable capacitated CPP by embedding the capability to predict the controller failures in forwarding devices. However, their proposed model is based on linear assignment of switches to controllers. Also, this approach aims to minimize the maximum, for all forwarding devices, of the sum of two latencies, namely, latency from the forwarding device to the first nearest controller and latency from the first nearest controller to its second closest controller. Also, the network search space is equivalent to NC bits due to the use of the decision variable r_{ij}^{l} if the linear assignment of a switch to l controller is expanded, which is similar to [2, 5]. Also, Killi and Rao [7] discussed the capacitated next controller placement (CNCP) problem by planning ahead the linear assignment of forwarding devices-controllers and considering no controller failure, one controller failure and multiple controller failures. The purpose is to minimize the maximum latency from all forwarding devices to the second reference controller in case of failure of the first controller assigned to it. Their mathematical formulation is in the form of three formulations (the first formulation with two-indexed variables using the variables r_{ij}^{1} and r_{ij}^{2}, the second formulation with the three-indexed variable r_{ijk} for one controller failure, and the third formulation with the three-indexed variable r_{ijk}^{l} to extend the failure to multiple controllers), in which the network search space, assuming N switches and C controllers, is equivalent to $2NC$ bits.

Research gaps in the present study are as follows: (1) Incorrect linear assignment of the next controller to the switches according to the next p-center problem in case of failure of the primary controller which leads to a significant increase in latency, and (2) Non-use of mathematical formulations of related works of binary decision variables that increase the network search space. Given the above research gaps, the key motivation for doing this research is to customize the next hierarchical p-center problem with a slight modification to the network real world in order to minimize the newly defined objective function. Also, we seek to reduce the network search space without losing all key locations so that the optimizer and heuristic algorithm used in this paper lead to the optimal solution faster. In other words, reducing the network search space is not to find the optimal solution, but this reduction approach has been used to quickly converge to the optimal solution. This paper examines the following cases: First, the Star Capacity-aware Latency-based Next Controller Placement Problem (SCLNCPP) in SD-WANs is formulated with array decision variables. It not only considers the capacity and reliability of the controllers, but also plans ahead for the controller failures (to assign the switches to the controllers

in star form). Also, the new optimal formulation retains all positions in the array due to the use of array variables and significantly reduces the network search space. This approach is used for delay-sensitive and more flexible applications. Also, instead of linearly assigning each switch to the first controller nearest to the switch and the second controller nearest to the first controller, we, inspired by the star next k-center problem, assign the switch to the first nearest controller and the second closest controller to the same switch. We formulate the SCLNCPP as a MIP using a combination of standard array two-indexed and three-indexed variables. The purpose of the proposed approach is to minimize the maximum, for all forwarding devices, of the sum of the latency from the switch to the nearest first controller with enough capacity and the latency from the same switch to the closest second controller with enough capacity. Second, due to the long running time of solving linear programming problems with CPLEX optimizer, we solve the proposed problem effectively on large-scale networks with a population-based simulated annealing algorithm. Finally, a detailed analysis of SCLNCPP is provided for different real-world topologies under different settings and compared with CNCP [7] and RCCPP [2].

The rest of the paper is organized as follows: In Sect. 2, we review previous researches, including classification of CPPs, capacity-aware resilient CPPs, and visualization-based resilient CPPs. The network model and problem formulation are described in Sect. 3, which includes problem formulation for one failure and its expansion for multiple failures. In Sect. 4, to solve the proposed problem, we use population-based heuristic approaches, which are described in detail. In Sect. 5, estimation of the performance of mathematical formulation, heuristic solutions according to the simulation results, and optimizations performed are stated. The last and final Section draws the conclusion of the paper and provides future research directions.

2 Related researches on CPP

2.1 Overview of different CPP classifications

According to [2], there are important factors for placing controllers in SDN-based network, which are: (A) the switch-controller latency [8], which is the sum of transmission delay, processing delay, and propagation delay; (B) the inter-controller latency [9], which is important for synchronization purposes in the case of having multiple controllers assigned to one forwarding device or for inter-domain controller communications; (C) the capacity of the controllers [10], which is usually defined as the number of requests (packets) of flow matching per second to be able to manage traffic demands; (D) traffic demands of forwarding devices, which shall be considered in order to avoid network congestion and overload of controllers; (E) scalability; and (F) resilient CPP, which is a type of CPP in SDN that emphasizes optimization of various aspects of reliability of control plane, increase in fault tolerance, and so on. Also, the CPP can be divided into three categories depending on the type of operation:

- Coordinated approaches [11]
- Phased approaches [12]
- A combination of the above two approaches.

Coordinated approaches find locations of controllers and control paths simultaneously using linear programming or mathematical tools. However, since the CPP is an NP-hard problem [12], these approaches may take much longer to search the entire solution space when the scale of the network is large. Unlike coordinated approaches, phased approaches divide the entire SDN into several control domains, and then one controller places in each domain. In each control domain, they select the appropriate control paths to connect each forwarding device and its controller. Hybrid approaches, in fact, consider both coordinated approach and phased approach to the CPP. Our formulation follows a mixed integer programming approach, and even problem-solving heuristic algorithms are used to reduce search space time. You can also find different categories of CPPs in [13], including CPPs with/without considering controller capacity, even CPPs based on statics and dynamics of network traffic, network element failure-aware CPPs, and partitioning-based CPPs. Our proposed approach is capacitated, static traffic-aware, controller failure-aware, and non-partitioning.

2.2 Overview of CPPs based on switch-to-controller assignment decision variable

Fan et al. [14] formulated the CPP with the purpose of minimizing communication delay, with and without the failure of a particular single link. In their mathematical formulation, they used the variable $x_{i,k}$ to assign forwarding devices to controllers, in which network search space, due to this variable, is estimated *NC* bit. They also provided a delay-aware reliable controller placement algorithm to solve the problem. In their paper, the assignment of each switch to multiple controllers was not linear, and a compromise between the primary path propagation delay and the backup path propagation delay was investigated. Killi and Rao [15] investigated the Capacitated Controller Placement Issue (CCPI) with the purpose of recovering single link failure, without a significant increase in the worst-case forwarding device-to-controller latency (minimizing the worst-case forwarding device-to-controller latency regardless of estimated capacity constraints). They provided a mathematical formulation that considers both the worst-case latency with and without link failure in the objective function of the proposed model. Also, in their mathematical formulation, they used the variable $x_{i,k}$ to assign forwarding devices to controllers, in which network search space, according to variable, is estimated *NC* bit. To solve the proposed model, they have shown that hierarchical controller placement approach for single link failure not only reduce the worst-case forwarding device-to-controller latency, but also reduce the maximum and average inter-controller latency.

Another study is based on Capacitated Next Controller Placement (CNCP) [7], in which a resilient controller placement approach was presented despite considering controller capacity and controller failures. The inter-controller latency and the number of controllers were considered as a budget. However, they linearly assigned a

forwarding device to a primary (master) controller and a backup (slave) controller. The purpose was to minimize the worst-case latency in case of controller failures. Killi and Rao's mathematical formulations are divided into three categories: 1) Problem formulation with two-indexed variables; 2) Problem formulation with three-indexed variables; and 3) Expansion of problem formulation with three-indexed variables for multiple controller failures. In the first, second, and third formulations, they used the assignment decision variables r_{ij}^{1} and r_{ij}^{2}, $\mathrm{r_{ijk}}$ and r_{ijk}^{l}, respectively (in fact, i denotes the switch number, j and k the controllers number, and l the controller failure number). According to the variables used, the estimated network search space, assuming C controllers and N switches, is $2NC$ bits. In addition, the proposed formulations are derived from the hierarchical next k-center problem. Also, to solve the CNCP, a simulated annealing heuristic algorithm is used to achieve the optimal value. Despite the similarities with CNCP, our formulation differs from this approach. First, the CNCP formulation is derived from the linear next k-center problem, but our formulation is derived from the customized star next k-center problem for a network real world, which changes the objective function; that is, in CNCP, each switch is assigned to the first nearest controller, and also if the first controller fails, the switch is assigned linearly to the second controller, which is the closest controller to the first switch controller. In our proposed problem, each switch is assigned to the first nearest controller, but if the first nearest controller fails, the same switch is assigned to the closest controller (as the second controller) in star form. Second, in CNCP, two-indexed and three-indexed assignment decision variables are used, which leads to a very large network search space ($2NC$ bits), but in our formulation, due to the use of array variables, network search space is reduced significantly (at most $\frac{NC}{2}$ bit). Finally, to solve our proposed problem, we use the population-based simulated annealing heuristic algorithm, which leads to not getting caught up in local optimum and fast convergence to global optimum, and low computation time to find a reasonable solution, while to solve CNCP a solo-searcher stimulated annealing algorithm was used, which has two main disadvantages of getting caught up in local optimum and high computation time to find an appropriate solution. In [16], an approach to deploying controllers is presented that minimizes the maximum load imbalance between controllers despite implementation of balancing and placement constraints. The authors used the decision variable $A_{s,c}$ to assign exactly one controller to the forwarding device, which leads to the NC bit network search space. Their scalable algorithm uses poly-stable matching to distribute a fraction of the forwarding devices evenly between the controllers to calculate near-optimal solutions. The purpose is to minimize load imbalances on large-scale networks. Also, in this research, the remaining switches are assigned to their nearest controller, despite considering the latency of the forwarding devices and the load of the controllers. The resilient deployment of the controller is directly related to node failure or link failure problems. In [17], a Hybrid Particle Swarm Optimization and Simulated Annealing (HPSOSA) meta-heuristic algorithm for resilient controller placement was proposed to minimize the average latency. Also, in their mathematical formulation, they used the decision variable z_{ijk} to assign the forwarding device to its first and second controllers, in which network search space is equivalent to $2NC$ bits.

The number of controllers and their locations in a network may cause delays to the forwarding devices, and the value of the worst-case latency parameter increases with single link failure. In order to prevent the increase in the introduced metric, planning is required for failure. In [18], a meta-heuristic approach called Capacitated Controller Placement Grey Wolf Optimization (CCPGWO) was proposed to search for the best location of controllers in a network so that the worst-case latency metric does not increase drastically despite link failure. Also, in their mathematical formulation, the authors used the decision variable b_{ij} to assign the forwarding device to its r controller, in which network search space is equivalent to NC bit. The formulations of [17, 18] are derived from the linear next k-center problem.

In [2], Tanha et al. examined the Resilient Capacitated Controller Placement Problem (RCCPP) with the purpose of minimizing the number of controllers in order to have a cost-effective design by considering only the failure of one or more controllers. Also, in this study, the constraint related to the resilience of the control path was added to the problem formulation. The authors offered a solution to this problem that, unlike previous studies, considers the parameters of switch-controller latency, controller-controller latency and controller capacity to estimate the traffic load of forwarding devices. They also offered two heuristic algorithms to solve the problem: RCCPP with all maximal cliques and RCCPP with single maximal clique. The RCCPP heuristic algorithm with all maximal cliques has exponential time complexity, which is not desirable in practical settings and for large-scale networks. For this purpose, they used the RCCPP algorithm with single maximal clique, which is a polynomial-time algorithm that uses the clique-based approach in graph theory, and have heuristically found high-quality solutions. The mathematical formulation of this approach is also non-optimal due to the use of non-standard binary decision variables, and the search space of this approach is reduced compared to the study of Killi and Rao [7]. With our proposed approach, we significantly reduced the network search space compared to the studies of Killi and Rao [7] and Tanha et al. [2]. For instance, if the network is assumed to have N forwarding device (switch) and C controller, first, the decision variable used to formulate this approach is x_{ij} (this variable is equal to one if switch i is connected to the controller at node j, otherwise it is equal to zero). Also, in this research, in order to create resilience, the number of controllers that serve a certain switch is equal to r. As a result, the network search space is equal to NC with respect to the decision variables used above, which is reduced by half compared to the study of Killi and Rao [7]. In our optimal formulation approach, using the network search space array decision variable, it is reduced almost four times compared to the study of Killi and Rao [7] and two times in comparison to the research of Tanha et al. [2]. Second, this approach is not suitable for delay-sensitive applications because its formulation is based on the k-center or k-medium facility location problem, which is more focused on cost, but our proposed approach and the research of Killi and Rao [7] are appropriate for delay-sensitive applications. The use of clique-based polynomial algorithms to solve the problem offers high-quality heuristic solutions, but it seems that injecting the concept of population into the heuristic algorithm in our proposed approach provides much better results.

Killi and Rao [5] proposed an optimization model for controller placement and predetermined mapping of forwarding devices to controllers, despite the high resilience against controller failure. They reduced the cost of installing controllers on the network, while guaranteeing the mapping of forwarding devices to predefined controllers. However, they provided other optimization models to reduce the reserved backup extra capacity of controllers, while keeping the number of controllers required constant. The authors used the decision variable, M_{ij}, to assign each forwarding device to r controller in star form, in which network search space is equivalent to NC bits. In fact, the superiority of our proposed approach over this goes back to the array decision variable, which reduces the network search space at best to $\frac{NC}{2}$ bit (by selecting the minimum number of controllers and minimizing the latency parameter). Table 1 shows the superiority of the proposed approach over the delay-sensitive and cost-sensitive approaches.

3 Network model and problem formulation

3.1 Basic problem and its correspondence to software-defined wide-area networks

Published papers on Capacitated Controller Placement Problem (CCPP) and its various forms considering latency, resilience and cost parameters based on k-center [19], k-means [20], and facility location problem [21], are considered as NP-hard issues [4]. The above issues, in their discrete state in the network search space, pursue a specific concept and purpose; for instance, in the k-center problem, we have a limited set of demand points (users) for services and a limited set of supply points (cities) to place warehouses, which aims to minimize the maximum distance from a demand point to the nearest corresponding supply point. The Star Capacity-aware Latency-based Next Controller Placement Problem (SCLNCPP) corresponds to the star next k-center problem with an optimal network search space compared to previous studies, which is NP-hard for both Euclidean and rectangular metrics [22]. The purpose of the next k-center problem is the same as the stated purpose for k-center, but in the next k-center, in case of failure of the first embedded supply points, the demand points are assigned to the second nearest corresponding supply points. We also customize the linear next k-center problem and convert to the star next k-center problem in order to better represent the network real-world applications. Given the above proposed problem, controllers play the role of warehouses and forwarding devices (switches). In other word, it plays the role of customers in a software-defined wide-area network.

Table 1 A comparison between the proposed method and other related works

Research	Placement Metrics	Approach Type (Coordinated, Phased or Both)	Decision Variables for Assigning Switches to the First and Second Controllers	Network Search Space Assuming N switches and C controllers	Advantages and Disadvantages of the Approach	Solutions/Algorithms
Fan et al. [14]	Delay-aware Resilient Controller Placement Problem	Phased approach (partitioning)	x_{vu}	*NC bits*	Advantage: To minimize the latency of primary paths and backup paths and make a compromise between the two latencies Disadvantage: No coverage of failure of more than one link, ignoring cost, large network search space	Mathematical Model of Exponential Decay Failure/Min-Cut-Based Controller Placement Algorithm by Binary Search
Killi and Rao [15]	Delay-based Link Failure-aware Controller Placement Problem	Coordinate approach	x_{ij}	*NC bits*	Advantage: To minimize the worst-case latency between switches and controllers Disadvantage: Taken from the linear k-center problem, no coverage of failure of more than one link, large network search space	Mixed Integer-Linear Programming (MILP) Mathematical Model

Table 1 (continued)

Research	Placement Metrics	Approach Type (Coordinated, Phased or Both)	Decision Variables for Assigning Switches to the First and Second Controllers	Network Search Space Assuming N switches and C controllers	Advantages and Disadvantages of the Approach	Solutions/Algorithms
Killi and Rao [7]	Delay-aware Resilient Capacitated Next Controller Placement Problem	Coordinated approach	x_{ijk} ‘ x^{J}_{ijk} ‘ x^{1}_{ij} x^{2}_{ij}	$2NC$ bits	Advantage: To minimize the maximum latency from all switches to its first and second controllers Disadvantage: Incorrect definition of the linear next (linear k-center) controller, very large network search space	MILP Mathematical Model/simple and solo-searcher simulated annealing meta-heuristic algorithm
Killi and Rao [16]	Delay-based Load Balance-aware Controller Placement Problem	Coordinated approach	$A_{s,c}$	NC bits	Advantage: To establish optimal load balancing between controllers, to minimize latency between switches and controllers Disadvantage: Reliability metric is not considered	Nonlinear mathematical model based on poly-stable matching/scalable algorithm with load balance constraints

Table 1 (continued)

Research	Placement Metrics	Approach Type (Coordinated, Phased or Both)	Decision Variables for Assigning Switches to the First and Second Controllers	Network Search Space Assuming N switches and C controllers	Advantages and Disadvantages of the Approach	Solutions/Algorithms
Kanodia et al. [17]	Delay-based Controller Failure-aware Controller Placement Problem	Coordinated approach	z_{ijk}	$2NC$ bits	Advantage: To minimize the average latency from all switches to the first controller and the second controller, Disadvantage: Incorrect definition of the linear next (linear k-center) controller, very large network search space	MILP/ hybrid particle swarm optimization and simulated annealing meta-heuristic algorithm
Kanodia et al. [18]	Delay-based Link Failure-aware Controller Placement Problem	Coordinated approach	b_{ij}	NC bits	Advantage: To minimize the worst-case delay in case of link failure Disadvantage: Ignoring controller failure and network cost	MILP/Grey Wolf Controller Placement

Table 1 (continued)

Research	Placement Metrics	Approach Type (Coordinated, Phased or Both)	Decision Variables for Assigning Switches to the First and Second Controllers	Network Search Space Assuming N switches and C controllers	Advantages and Disadvantages of the Approach	Solutions/Algorithms
Tanha et al. [2]	Cost-based Controller Placement Problem	Coordinated approach and phased approach	x_{ij}	NC bits	Advantage: To minimize the number of controllers Disadvantage: Unsuitable for delay-sensitive applications, no coverage of failure of multiple controllers or multiple links	Mathematical model/clique-based polynomial algorithm
Killi and Rao [5]	Cost-based Resilience-aware Controller Placement Problem	Coordinated approach	M_{ij}	NC bits	Advantage: To minimize cost and to make a compromise between cost and resilience metrics Disadvantage: Very large network search space, ignoring latency metric	Optimal mathematical model
Proposed approach	Star Capacity-aware Latency-based Next Controller Placement Problem (SCLNCPP)	Coordinated approach and phased approach	$AP^{[v],s}_{[u]\left[\overline{\overline{u}}\right]}$	At most $\frac{NC}{2}$ bits and at least $N\hat{C}$ bits (2 N N_b bits)	Advantage: To significantly reduce network search space, to significantly minimize defined latency metric	Mixed Integer Programming (MIP) Model/population-based simulated annealing algorithm

3.2 Proposed approach and its assumptions

3.2.1 Ideas of proposed approach

Considering the disadvantages of the previous researches mentioned in Sect. 2.2, in this research, we tried to reduce the network search space without losing the original and acceptable locations, by using standard array variables in mathematical formulation. Therefore, we can optimally perform global search (exploration) and extraction (exploitation) operations with solvers and problem-solving heuristic algorithms. Exploration seeks global search, that is, it provides new solutions, while extraction seeks small and important changes that make changes on the current solution. The reason for this long time is that more search is needed, that is, the act of detecting. In extraction, existing array solutions are utilized to achieve more effective operations. Therefore, a tradeoff between these two factors seems necessary. In fact, minimizing the network search space without losing key locations helps the exploration (global search) a lot, so that the CPLEX optimizer and the heuristic algorithm used can reach the solution quickly (not that the goal is the optimal solution, but the goal is to quickly converge to the solution). In this section, for the new mathematical formulation of the star capacity-aware resilient delay-based next CPP, instead of the decision variables used in the study of Killi and Rao [7], namely, r_{ij}^1, r_{ij}^2, r_{ijk} and r_{ijk}^l, and the decision variable x_{ij} in the study of Tanha et al. [2], and other previous researches performed ([5, 14–18]), we use the decision variable A as a two-dimensional array if one controller fails or multidimensional array if the formulation is expanded to multiple controller failure. The two-dimensional array A and the multidimensional array A in case of failure of one or more controllers are shown in Figs. 1 and 2, respectively. Actually, in the first case (Fig. 1), the array is a two-dimensional matrix, with n rows and 2 columns. Also, in the second case (Fig. 2), the array is a multidimensional matrix, with n rows and $m > 2$ columns.

In the $A^1_{[v][u]}$ two-dimensional array, according to Fig. 1, the row of the array represents the number of switches in the network, n, the first column of the array indicates the first controller of switches 1 to n, and the second column of the array

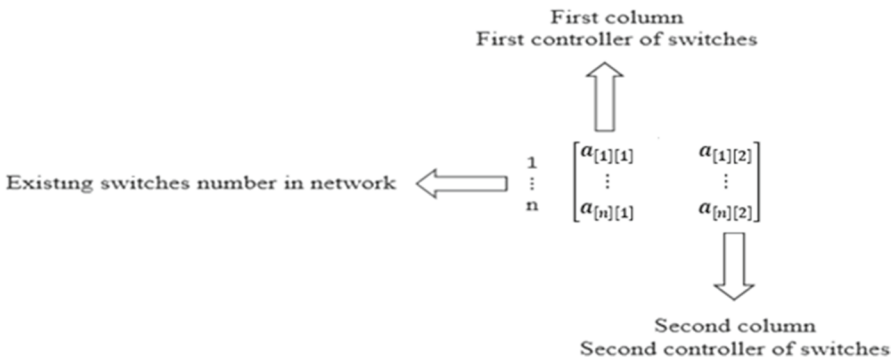

Fig. 1 Two-dimensional array $A_{n\times 2}$(n is the number of switches and 2 is the number of controllers assigned to the switches) in case of one controller failure

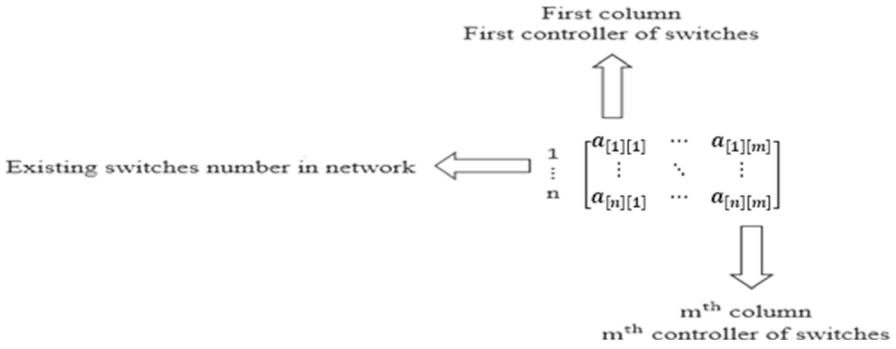

Fig. 2 Multidimensional array $A_{n\times m}$(A is a matrix of dimensions $n\times m$ where n is the number of switches and m is the number of controllers assigned to the switches, $m=F+1$ where F is the number of controller failures and the entries of the matrices are numbers {0, …, M-1} where M is the number of controllers) in case of multiple controller failure

shows the second controller of switches 1 to n. We specify the switch with index v and the controller with index u. Also, in the above two-dimensional array, we allocate two first and second controllers to each of the switches in the network (from 1 to n). In addition, in the $A^F_{[v][u]}$ multidimensional array, all of the above is true, but in this array, a switch v is assigned to a list of m controllers u. The number of failures extends from one controller failure to multiple controller failures (F represents different failure scenarios). We first state the assumptions on the $A^1_{[v][u]}$ two-dimensional array and then extend the above-mentioned cases to the $A^F_{[v][u]}$ multidimensional array. In the $\mathrm{A}^1_{[v][u]}$ two-dimensional array, the first assumption is that two repetitious controllers cannot be assigned to the switch in the first row and in the first and second columns. In other words, **i**n every row, there cannot be multiple occurrences of the same controller. Also, considering the maximum number of controllers placed in the SD-WAN with the variable MAX(M), the second assumption is that the values in the first and second columns of the array can be numbers between 1 and the maximum number of controllers MAX(M). The third assumption is based on the fact that the minimum number of controllers allocated to switch i can be 3 in the minimum case because we consider the number of controller failures in the network to be two failures. In other words, $m=F+1$ where F is the number of failures considered. Also, we will try to pursue a cost-effective network approach by placing fewer controllers. In the proposed approach idea, we calculate the number of bits per array column according to Formula (1), given the maximum number of controllers MAX(M). In Formula (1), the *ceil* function always rounds the value of the function to the upper bound. According to the previous example, if we have N switches in a network and want to place the M controllers in the network, we calculate the number of bits of the network search space as Formula (2):

$$\mathrm{N_b} = \mathrm{ceil}(\log_2 \mathrm{MAX(M)} \tag{1}$$

$$\begin{aligned} &\text{Number of bits for switches} \\ &\times (\text{number of bits in the first column} \\ &+ \text{number of bits in the second column}) \end{aligned} \tag{2}$$

We denote the number of bits in the first column or the second column by variable N_b. Here, if the problem of placing one controller in the network is raised, network resilience is violated. However, to make the network resilient, we would attempt to assign one switch to multiple controllers. According to the above formula, the number of bits of the network search space in the new mathematical formulation will be equal to $2\ N_b N$ bits. As a numerical example, if the number of switches and controllers in an SD-WAN is 112 and 11, respectively, the number of bits of the network search space of the proposed approach will be equal to 896 bits, which reduces the network search space by about 1.3 to several times compared to the previous studies. The same value is equal to 2464 bits for Killi and Rao work, assuming the values of the same example, and 1232 bits for Tanha et al. work.

Also, in the next step, in the second approach proposed in this paper, an attempt is made to assign subsequent controllers to the switches in star form, by customizing the next k-center problem. We customize the linear k-center problem in such a way that we first take it out of emergency applications and adapt it to the network real world in star form by changing the assignment of each switch to the first controller and the second controller. Second, we assume that the switch is notified of the controller failure or non-failure (the controller placed on the next nearest switch) by sending $packet - in$ messages. In the intended network, if the controller

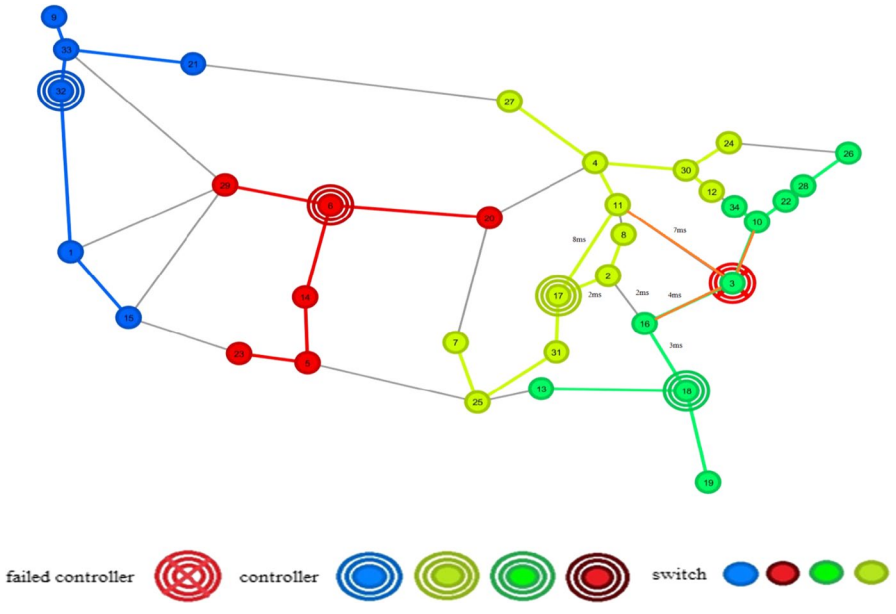

Fig. 3 An example to illustrate the customized next k-center problem

is deployed on the same switch, here the capability to detect failure is deterministic. As an example, we describe the disadvantages of the next linear k-center problem as shown in Fig. 3, and describe our proposed approach as a customized next k-center problem. In Fig. 3, five controllers are deployed in a small-scale network of Internet 2 OS3E topology, and the color and shape of the switches assigned to the controller are similar. We illustrate our proposed approach in some part of Fig. 3 so that only in part, the numbers on the link indicate the hypothetical delay between the network elements.

Usually, in previous researches, the assignment of switches to controllers is linear with respect to the next k-center problem in case of failure of one or more controllers. In Fig. 3, switch No. 11 is assigned to its first nearest controller with enough capacity (controller No. 3 with a delay of 7 ms). Also, switch No. 11, in case of failure of controller No. 3, according to the planning ahead for controller failures, is assigned linearly to the second closest controller with enough extra capacity (controller No. 18 with a delay of 7 ms to the first controller and a delay of 14 ms to switch No. 11). Although previous studies have considered the linear pre-assignment planning of switches to the controllers for controller failure, and even in case of failure, when reassigning switches to their respective controllers, they experience an increase in the worst-case delay; therefore, by customizing the next k-center problem for the real-world network, we seek to plan the star pre-assignment of the switches to the second nearest controllers with enough capacity to prevent disconnections and dramatic increase in the worst-case latency in a case of the first controller failure. For instance, in Fig. 3, switch No. 11 does not need to move to controller No. 3 similar to emergency applications, and by sending a $packet - in$ message to it and not receiving the $packet - out$ message in the network real world, it detects the failure of controller No. 3. However, instead of being assigned to the second closest controller with enough capacity (controller No. 18 with a delay of 14 ms) to the first controller (controller No. 3), switch No. 11 is assigned to its second nearest controller with enough extra capacity (controller No. 17 with a delay of 8 ms); therefore, in addition to adapting the problem to the network real world, with this new approach, we will reduce the worst-case latency between switches and controllers. As a result, the two proposed ideas of reducing the network search space by defining the standard array variable in the new mathematical formulation and customizing the linear next k-center problem in star form simultaneously will show a significant improvement in the metrics studied.

3.2.2 Description of controller failure and solution of reassignment problem in SD-WANs

The amount of failure in SD-WANs varies, so that the failure may be equal to the failures of the controller node and the failures of the controller site. Firstly, controller site failures occur less frequently than controller node failures, and if this happens, we should consider having a backup data center elsewhere; secondly, as a result of an attack or a disaster, the data centers where the controllers are located are completely destroyed; therefore, in order to establish fault tolerance in the network,

we assign two controllers to each switch, in the form of array mathematical formulation, for the failure of one controller. Despite failures of two controllers in the network, we have to assign three controllers to each switch, and thus for the failure of F controller, we have to assign M controllers to each switch. We follow the master/slave model as OpenFlow v1.2 and higher versions [23] in addition to providing a simple fail-over mechanism (including definition of a list of controllers for the switch) using greedy fail-over methods [24] allow a controller to play one of the three roles of master, equal and slave for the switch. The master or equal controllers receive packet − in messages sent by the switches, so that the first controller of each switch is the master controller (with full control over the switch) and the other controllers of the same switch are the slave controllers (with read-only access to the switch). Also, we need to coordinate the distributed controllers in order to select the master controller [9], which is only limited to constraint in the mathematical formulation of the problem and will not be emphasized in detail.

Failure of one or more controllers in the SD-WAN may cause disconnections between the controllers and the forwarding devices. The $\mathrm{FC} \subset \mathrm{M}$ parameter is a set of failed controllers due to hardware or software issues. Disconnected switches in case of failure of connections between switches and controllers ($\overline{N} \subset N$) need to be reassigned to one of the open or active controllers in $\mathrm{OC} = \mathrm{AC} = \mathrm{M} - \mathrm{FC}$ network with enough capacity. The network administrator can connect each forwarding device of the failed controller to an active (open) controller with enough extra capacity by solving the assignment problem according to Formulas (3) to (6).

$$\min\{\mathrm{LAT}_{\mathrm{total}}\}$$

Subject to

$$\sum_{\overline{\overline{u}} \in \mathrm{AC}} \mathrm{AP}^{\mathrm{F}}_{[v]\left[\overline{\overline{u}}\right]} = 1 \forall \mathrm{v} \in \overline{\mathrm{N}} \tag{3}$$

$$\sum_{v \in \overline{\mathrm{N}}} \mathrm{load}_{[v]}.\mathrm{AP}^{\mathrm{F}}_{[v]\left[\overline{\overline{u}}\right]} \leq \overline{\mathrm{U}_{\left[\overline{\overline{u}}\right]}}.\mathrm{y}_{\left[\overline{\overline{u}}\right]} \forall \overline{\overline{u}} \in \mathrm{AC} \tag{4}$$

$$\mathrm{LAT}_{\mathrm{total}} \geq \sum_{\overline{\overline{u}} \in \mathrm{AC}} \mathrm{l}_{[v]\left[\overline{\overline{u}}\right]}.\mathrm{AP}^{\mathrm{F}}_{[v]\left[\overline{\overline{u}}\right]} \forall \mathrm{v} \in \overline{\mathrm{N}} \tag{5}$$

$$\mathrm{AP}^{\mathrm{F}}_{[v]\left[\overline{\overline{u}}\right]} \in \{0, 1\} \forall \mathrm{v} \in \overline{\mathrm{N}}, \forall \overline{\overline{u}} \in \mathrm{AC} \tag{6}$$

In the above equations, F represents the various failure scenarios, v, switch index, u, $\overline{u}$, and $\overline{\overline{u}}$, the controller index. Constraint (3) guarantees the assignment of the forwarding device (switch) to a unique active controller in the two-dimensional array RP, despite the various failure scenarios of F controller (v is the number of the disconnected switches in case of failures and $\overline{\overline{u}}$, is the number of the active controller in the network). Constraint (4) makes it possible for the total demand of

disconnected switches to an active controller $\overline{\overline{u}}$, in the two-dimensional array RP not to exceed the remaining capacity of the active controller; ($\text{load}_{[v]}$ denotes the total demand of the switches in each row of the array from the controllers assigned to them and $\overline{U_{[u]}}$ denotes the remaining capacity of the active controller $\overline{\overline{u}}$, in the array). Therefore, the purpose is to assign forwarding devices of failed controllers to other active controllers with enough extra capacity in order to minimize the worst-case delay according to constraint (5). In fact, in constraint (5), $l_{[v]\left[\overline{\overline{u}}\right]}$ is the minimum delay between forwarding device v and the active controller $\overline{\overline{u}}$ in the array and LAT_{total} is the objective function. However, in order to avoid a significant increase in latency due to failures that occur, we need to plan ahead controllers in star form by customizing the linear next k-center problem. Also, for the problem of reassigning the switch to the active controllers, the decision variable $AP^{F}_{[v]\left[\overline{\overline{u}}\right]}$ is used. This variable is equal to 1 if $\overline{\overline{u}}$ is the active controller assigned to switch v in the two-dimensional array RP assuming $F = 1$ failure, otherwise it is 0. Also, the variable $y\left[\overline{\overline{u}}\right]$ is equal to 1 if an active controller is deployed on the switch $\overline{\overline{u}}$ in array modeling, otherwise it is 0.

3.2.3 Purpose of hybrid optimization approach

Our proposed problem, like [7], is for delay-sensitive applications and is not cost-based (minimizing the number of controllers in the network). Nevertheless, we allow the network designer to select and adjust the service quality parameters (switch-controller latency and inter-controller latency) so that the objective value (sum of the latency from all switches to the first controller and second controller) is minimized. Depending on the requirements of the network, forwarding devices (switches) on the data plane always send $packet - in$ messages to their closest controller with enough extra capacity, regardless of whether it is available or not. Therefore, as soon as the $packet - in$ messages arrive, the switches send the packets to the first controller if it exists, otherwise they send them in star form to the closest controller of the same switch as the second controller. The notation $\text{LAT}_{\text{total}}$ aims to minimize the maximum delay, for all switches, of the sum of the latency from the forwarding device to the first nearest reference controllers and the latency from the same forwarding device to the second closest reference controller of the list of controllers, which is according to Formula (7):

$$\min_{N_c \subseteq L, \left|N_c\right| = l} \{LAT_{total}\} = \min_{\substack{N_c \subseteq L \\ \left|N_c\right| = l}} \max_{v \in \mathrm{N}} \left\{ \min_{u \in L} l_{[v][u]} + \min_{\substack{\overline{u} \in L \\ \overline{u} \neq u}} l_{[v][\overline{u}]} \right\} \tag{7}$$

In (7), N_c is a parameter supplied by the designer of the network, whose value is decided by the level of planning.

3.2.4 Bounds for Switch-Controller Latency and Inter-Controller Latency

Two latency parameters, namely the switch-controller latency and the inter-controller latency are effective in the mathematical formulation of the proposed problem. The latency between the forwarding device to the first nearest reference controller and the latency between the same switch to the second closest reference controller is guaranteed in case of failure of the first reference controller in the objective function and essence of the problem. However, the inter-controller latency parameter must be specified in the formulation constraints so that the distances between the controllers do not exceed the threshold defined for this metric. Despite the competition between the two switch-controller latency and inter-controller latency metrics, connecting switches to their first and second nearest controllers may lead to load imbalances between controllers and, as a result, we may subsequently experience an increase in latency due to queuing delay time on some controllers [25]. This paper does not discuss load imbalance metrics.

3.3 Formulation of Proposed Problem with Two-indexed Variables

The network graph $G(V, E)$ is represented by assuming F failure scenarios in the SD-WAN, where $V = N \cup L$, N is a set of OpenFlow switches, and L is a set of potential controller locations. Also, E denotes a set of weighted links. Link weight indicates the propagation delay between nodes. Also, let $C = \{c_1, c_2, \ldots, c_l\}$ be a set of controllers that can be deployed in the network. Assuming all switches are OpenFlow-enabled, however, they act as potential locations for the controller deployment (in fact, the controller is installed and implemented on the server and deployed inside the rack next to the switch), i.e., $N{=}L$. In fact, we store the network graph in a two-dimensional array A or multidimensional array A according to the number of failures and different failure scenarios. Also, it is assumed that if the number of failures in the network is one, the array decision variable AP will be a two-dimensional array. It will consist of the first reference controller and the second reference controller for each switch in each row of the array, otherwise the array AP expands to several dimensions. It is assumed that almost one controller fails at any time in the two-dimensional array AP. Also, we assume that there should be no repetitious values in each row of the array AP, because a switch must be connected to a non-repetitious first reference controller and a repetitious second reference controller in order to have a fault tolerance, so that in case of failure of the first controller, the switch will be assigned to its second controller.

In the new mathematical formulation of the proposed approach, it is assumed that the decision variables used are $y_{[u]}$, $AP^1_{[v][u]}$, $AP^2_{[v][u]}$ and $w_{[u]\left[\frac{=}{u}\right]}$, where v is a member of N, and u is a member of M. The variable $y_{[u]}$ determines whether a controller is deployed in site u or not. Note that the variable is equal to one if a controller is deployed at site u. For the switch v, the variable $AP^1_{[v][u]}$ is set to one if u is the first controller of forwarding device v in the two-dimensional array AP, otherwise it is zero. The variable $AP^2_{[v][u]}$ is set to one if u is the second controller of forwarding

Table 2 Decision variables used in the formulation of SCLNCPP

Variable	Definition
$\mathbf{y_{[u]}}$	= 1, if a controller is deployed in the forwarding device u; 0, otherwise
$\mathbf{AP^1_{[v][u]}}$	= 1, if u is the reference controller of the forwarding device v in the array AP; 0, otherwise
$\mathbf{AP^2_{[v][u]}}$	= 1, if u is the second controller of the forwarding device v in the array AP; 0, otherwise
$AP^{[v]}_{[u]\left[\overline{\overline{u}}\right]}$	= 1, if in v row of the arrayAP, there is a repetitious number is in both sites u and $\overline{\overline{u}}$; 0, otherwise
$w_{[u]\left[\overline{\overline{u}}\right]}$	= 1, if the controllers are deployed in both sites u and $\overline{\overline{u}}$; 0, otherwise
$AP^{[v].s}_{[u]\left[\overline{\overline{u}}\right]}$	= 1, if the nearest controller to switch v in the array AP is $\overline{u}$ after s-1 failure of the nearest controllers to it; 0, otherwise

Table 3 Notations used in the SCLNCPP

Notation	Definition
$\mathbf{G(N, E)}$	The network graph including N switches and E weighted links
$\mathbf{\underline{A}}$	A two-dimensional or multidimensional array to store switches in the array row and controllers in the array columns according to the number of failures
$\overline{\mathbf{N}}$	The subset of connected switches ($\overline{\mathrm{N}} \subset \mathrm{N}$)
$\mathbf{N}$	A set of network elements (OpenFlow switches)
$\mathbf{M}$	Number of controllers
$\mathbf{L}$	A set of potential locations for deployment of controllers
$\mathbf{E}$	A set of weighted links (propagation delay of intended graph) between network elements
$\boldsymbol{C}$	A set of controllers for being deployed in the network
$\mathbf{Load}_v$	Traffic load of forwarding device v (number of packet − in messages generated by forwarding device v)
$\mathbf{U}_u$	Capacity of controller u
$\mathbf{l}_{vu}$	Minimum propagation delay between node v and node u
$l_{v\overline{\overline{u}}}$	Minimum propagation delay between node v and node $\overline{\overline{u}}$
$\mathbf{CCLB_{max}}$	Maximum inter-controller latency
$\mathbf{D}_A$	shortest longest path length (the diameter of the graph topology)
$\mathbf{D}_{min}$	The minimum length of the shortest path in the array A
$\mathbf{N}_b$	The number of bits in each entry of array A
$\boldsymbol{N_c}$	A variable for planning level of controller failure
$\mathbf{LAT_{total}}$	Maximum, for all forwarding devices, of the sum of the latency from the forwarding device to the first controller and the latency from the same forwarding device to its second controller

device v, otherwise it is zero. The variable $w_{[u]\left[\overline{\overline{u}}\right]}$ specifies whether the controllers are deployed in site u and $\overline{\overline{u}}$ or not. Note that the variable $w_{[u]\left[\overline{\overline{u}}\right]}$ is set to 1 if the controllers are deployed in both sites u and $\overline{\overline{u}}$. Also, in this study, it is assumed that Formula (8) must hold, where D_A (length of the shortest longest path) denotes the

diameter of the graph topology, which is considered as an array. Also, D_{min} indicates the minimum length of the shortest path.

$$D_{min} \leq CCLB_{max} \leq \mathrm{D_A} \tag{8}$$

Tables 2 and 3 thus show the decision variables and notations used in the proposed mathematical formulation of SCLNCPP.

The Star Capacity-aware Latency-based Next Controller Placement Problem (SCLNCPP) is formulated in accordance with constraints (9) to (25):

$$\min_{N_c \subseteq L, |N_c| = l} \left\{ LAT_{total} \right\}$$

Subject to

$$\sum_{u \in M} y_{[u]} = m \tag{9}$$

$$\sum_{\mathrm{u} \in \mathrm{M}} AP^{1}_{[v][u]} = 1 \forall v \in N \tag{10}$$

$$\sum_{\substack{u \in M \\ u \neq v}} AP^{2}_{[v][u]} = 1 \forall v \in N \tag{11}$$

$$AP^{1}_{[v][u]} + AP^{2}_{[v][u]} \leq y_{[u]}.U_{[u]} \forall v \in N, \forall u \in M \tag{12}$$

$$(w_{[u]\left[\bar{\bar{\mathrm{u}}}\right]} l_{[u]\left[\bar{\bar{\mathrm{u}}}\right]}) \leq CCLB_{max} \forall u, \bar{\bar{u}} \in M \tag{13}$$

$$w_{[u]\left[\bar{\bar{u}}\right]} = y_{[u]} y_{\left[\bar{\bar{u}}\right]} \forall u, \bar{\bar{u}} \in M$$

$$y_{[u]} + \sum_{\substack{\bar{\bar{\mathrm{u}}} \in M \\ l_{[v]\left[\bar{\bar{\mathrm{u}}}\right]} > l_{[v][u]}}} AP^{1}_{[v]\left[\bar{\bar{\mathrm{u}}}\right]} \leq 1 \forall v \in N, \forall u, \bar{\bar{\mathrm{u}}} \in M \tag{15}$$

$$y_{\left[\bar{\bar{u}}\right]} + \sum_{\substack{\bar{u} \in M \\ l_{[v][\bar{u}]} > l_{[v]\left[\bar{\bar{u}}\right]}}} AP^{2}_{[v][\bar{u}]} \leq 1 \forall v \in N, \forall \bar{u}, \bar{\bar{\mathrm{u}}} \in M \tag{16}$$

$$\sum_{v\in N} load_{[v]} AP^{1}_{[v][u]} + \sum_{v\in N} load_{[v]} AP^{2}_{[v]\left[\overline{\overline{u}}\right]} \leq U_{[u]}.y_{[u]} \forall u, \overline{\overline{u}} \in M \tag{17}$$

$$AP^{[v]}_{[u]\left[\overline{\overline{u}}\right]} = \begin{cases} 0, u \neq \overline{\overline{u}} (\text{if there is no duplicate number in first column and second column of v row for array } A) \\ 1, \text{Otherwise} \end{cases} \tag{18}$$

$$LAT_{total} \geq \left(l_{[v][u]} + l_{[v]\left[\overline{\overline{u}}\right]}\right)\left(AP^{1}_{[v][u]} + AP^{2}_{[v]\left[\overline{\overline{u}}\right]} - 1\right) \forall v \in N, \forall u, \overline{\overline{u}} \in M \tag{19}$$

$$y_{[u]} \in \{0,1\} \forall u \in M \tag{20}$$

$$AP^{1}_{[v][u]}, AP^{2}_{[v][u]} \in \{0,1\} \forall v \in N, \forall u \in M \tag{21}$$

$$w_{[u]\left[\overline{\overline{u}}\right]} \in \{0,1\} \forall u, \overline{\overline{u}} \in M \tag{22}$$

Constraint (9) specifies the deployment of exactly m controllers in the SD-WAN. Each switch has its own first and second controllers in each row of the array AP, respectively, according to constraints (10) and (11). Constraint (11) also ensures that the first controller of forwarding device v does not need to be distinct from the controller deployed in v, while the second controller of forwarding device v must be distinct from the controller deployed in v. However, constraint (12), despite controlling the validity of the assignments between the switches and the corresponding first and second controllers, determines that u is either the first controller or the second controller for switch v in the two-dimensional array A (according to subsequent constraints, if the first column array A is u for switch v, certainly the second controller can no longer be u and must be a controller other than u. Now, if u is the second corresponding controller of the switch v, then the first corresponding controller of the switch can be another one except u (e.g., $\overline{\overline{u}}$)). Also, regarding the validity of the assignments, the assignment of switches is not allowed to u when no controllers is deployed in location u in the two-dimensional array A ($y_{[u]}$ is zero); thus, in other words, both the decision variables $AP^{1}_{[v][u]}$ and $AP^{2}_{[v][u]}$ are equal to zero. The latency between each pair of active controllers in the network (first or second column of the two-dimensional array A for network switches) should be less than the maximum or bound of latency between the controllers $CCLB_{max}$ (allowable maximum inter-controller latency), which is met by constraint (13). If both the decision variables $y_{[u]}$ and $y_{\left[\overline{\overline{u}}\right]}$ are equal to one, the variable $w_{[u]\left[\overline{\overline{u}}\right]}$ is equal to one. Given the term $w_{[u]\left[\overline{\overline{u}}\right]} = y_{[u]} y_{\left[\overline{\overline{u}}\right]}$, we use the constraint (14) to assign switches to the controllers in

star form. In other words, in this formulation, the linear assignment of the switches to the controllers is excluded. Given the constraint (14), it is guaranteed that the variable $w_{[u]\left[\overline{\overline{u}}\right]}$ is equal to one when both decision variables $y_{[u]}$ and $y_{\left[\overline{\overline{u}}\right]}$ are equal to one.

In the proposed formulation, we are inspired by the closest assignment constraint introduced by Wagner and Falkson [26] for switches v in the intended topology so that failure occurs at the nearest controller of switch v or it does not occur; therefore, we use constraint (15) to ensure that the first controller of forwarding device v is the controller nearest to it. We also embed a constraint to ensure that switch v is assigned to the second nearest corresponding controller in star form in case of failure of the first nearest reference controller of switch$v$. The constraints related to (15) operate as follows: If a controller is installed in u ($y_{[u]}$ is equal to one), then assignment of forwarding device v to any controller away from v (i.e., $\overline{\overline{u}}$) instead of u is not allowed (i.e.,$\sum_{\substack{\overline{\overline{u}} \in M \\ l_{[v]\left[\overline{\overline{u}}\right]} > l_{[v][u]}}} AP^{1}_{[v]\left[\overline{\overline{u}}\right]} = 0$). Also, in constraint (16), the same conditions prevail to ensure the nearest assignment between a forwarding device and its second controller in star form in case of failure of the first controller of the switch, explicitly by correcting the problem of the linear next k-center to the star next k-center. In other words, assigning switch v to any second controller away from v (i.e.,$\overline{u}$) instead of $\overline{\overline{u}}$ is not allowed (i.e.,$\sum_{\substack{\overline{\overline{u}} \in M \\ l_{[v]\left[\overline{\overline{u}}\right]} < l_{[v][\overline{u}]}}} AP^{2}_{[v][\overline{u}]} = 0$). The demand constraint (17) ensures that the total request of the forwarding devices managed by a controller u does not exceed the capacity U_u of that controller in the two-dimensional array$A$. Constraint (18) guarantees that the first reference controller u and the second reference controller $\overline{\overline{u}}$ in the two-dimensional array A must be different. According to constraint (18), if there is no repetitious number in v row of the two-dimensional array$A$, the value of $AP^{[v]}_{[u]\left[\overline{\overline{u}}\right]}$ is equal to zero, otherwise it is equal to one. According to constraint (19), for each switch v, the objective value must be greater than or equal to the sum of the latency from forwarding device v to the first controller u and the latency from the same switch to the second nearest controller $\overline{\overline{u}}$. The term $AP^{1}_{[v][u]} + AP^{2}_{[v]\left[\overline{\overline{u}}\right]} - 1$ becomes positive in the left-hand side of the constraint (19) when both decision variables $AP^{1}_{[v][u]}$ and $AP^{2}_{[v]\left[\overline{\overline{u}}\right]}$ are equal to one; finally, the right-hand side of the constraint (19) is reduced to $l_{[v][u]} + l_{[v]\left[\overline{\overline{u}}\right]}$ if the previous terms are met, which is equipollent to the sum of the latency from forwarding device v to its first controller u and latency from the same switch to its second closest controller $\overline{\overline{u}}$. The right-hand side of the constraint (19) is negative in other cases. The decision variables defined in numerical constraints (20) and (22)

accept values of zero or one. Constraint (21) is an array numerical constraint that accepts values of zero or one.

4 A heuristic solution for solving problem

The mathematical formulation with two-indexed array decision variables presented in the previous section consists of a quadratic number of constraints and variables. As the network size increases, so does the number of constraints; and as a result, the running time also increases. In order to converge to the optimal and faster solutions, we present a heuristic algorithm in this section. Unlike the heuristic algorithms used to solve controller placement problems in [2, 5, 7, 14, 15] and [16], the simulated annealing algorithm [7] is well-known for finding global optimum of the problems that have a large search space and many local optimums. The main feature of this algorithm is that it accepts solutions that are worse than the current solution with some probability to avoid getting stuck in a local optimum. In order to significantly improve the performance of the simulated annealing algorithm, we inject the population concept into the simulated annealing algorithm. Therefore, in addition to solving the SCLNCPP with mathematical formulation in CPLEX optimizer, we use the population-based stimulated annealing algorithm [27]. However, the simulated annealing algorithm suffers from two main drawbacks of being trapped in local minimum and spending high computation time to find an optimal solution; therefore, any bad luck affects the nature of the results and a local minimum may be obtained instead of the global minimum. In addition to the above disadvantages, it works with one point and it is not possible to provide much variety with one point. The main feature of the population-based version is the same as that of the normal version of the algorithm. Another distinctive feature of the algorithm is that several simulated annealing algorithms work in parallel, that is, apparently several algorithms work with one member of the population, while this is not the case. Each of the algorithms takes a member of the population and each generates a solution in turn. Therefore, each algorithm selects the best solutions according to the original population from these solutions. In addition, in the intended algorithm, several neighbors are generated simultaneously for each member of the main population, and the best neighbors are selected according to the size of the main population to be compared with each member of the main population; thus, with the parallelization of several simulated annealing algorithms, comparing one member of the main population with the neighbors of another member of the main population creates interaction. In other words, the concept of the population-based simulated annealing algorithm is quite similar to the simulated annealing algorithm as follows: Assuming a population size equal to *nPop*, if in each iteration, each member of the current population simultaneously generates a, *nNeighbour*, neighbor, then, we will have $nPop \times nNeighbour$ new solutions. All generated neighbors are in one set. Then, the members of this set are compared with each other according to the rules of the algorithm and the best *nPop* members are selected according to the number of the main member population. This method ensures that each member of the population

interacts with all the neighbors generated by the other members of the main population, instead of just comparing it with its neighbors. Also, generated *nNeighbour* neighbors must be generated for each of the best selected *nPop* members, and the same process of generating *nNeighbour*-sized neighbors by the winning neighbors from among $nPop \times nNeighbour$ solution is repeated until the termination condition is met; thus, the pseudocode related to the population-based simulated annealing algorithm to solve the SCLNCPP is shown in Algorithm 1.

Algorithm 1 Population-based Simulated Annealing Algorithm

Input: G (N, E), $T_{initial}$, T_{end}, $Iter_{max}$, $SubIter_{max}$, $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$, nPop, nNeighbor
Output: SOL^*, OF^*
1. Temp = $T_{initial}$, nPop = **n**, nNeighbour = **m**, OF^*= inf, $\mathbf{It} = \mathbf{It_0}$
2. SOL= Pop_0 = GenerateRandomInitialPopulation ( )
3. OF = EvualuateRandomInitialPopulation (SOL)
4. **while** Temp ≥ T_{end} **do**
5. **for** $Iter_{max}$ = 1: MaxIt /Maximum iteration proportional to the size of the problem/
6. **for** $SubIter_{max}$ = 1: It /$\mathbf{It} = \mathbf{It_0}$ Number of iterations per temperature/
7. **if** Feasibility (SOL) == 1 and OF < OF^* **then**
8. SOL^*=SOL, OF^*=OF
9. **end if**
10. **for** i= 1: ***n***
11. **for** j= 1: ***m***
12. /Generate neighbors for each member of the population/
SOL′ = newPop (i, j) = GenerateRandomNeighbour (SOL))
13. OF′ = newPop (i, j) /Evaluate neighbors for each ***i*** and ***j***/
14. Pop_1 = newPop (i)
/Relative comparison of each member of the main population with one of the selectable members of OF′/
15. $\Delta = \frac{\mathbf{f(Pop_1)-f(Pop_0)}}{\mathbf{f(Pop_0)}}$
16. **if** $\Delta \leq \mathbf{0}$ **then**
17. SOL = SOL′= newPop(i)= Pop_1 /Accept the neighbor's solution/
18. **else**
19. Accept newPop(i) or Pop_1 with a probability $\mathbf{e}^{-\frac{\Delta}{\mathbf{Temp}}}$
20. **end if**
21. **end for**
22. **end for**
23. **end for**
24. Temp =$\boldsymbol{\gamma}$. Temp
25. It= $\frac{\mathbf{It}}{\boldsymbol{\beta}}$
26. **end for**
27. **end while**
28. **return** SOL^*, OF^*

In Algorithm 1, in step 1, the temperature value (i.e., *Temp*) is set to $T_{initial}$ as the initial temperature. In this algorithm, we set the starting temperature ($T_{initial}$) and the end temperature (T_{end}) to 15 and 4.99, respectively. Also, a geometric cooling function is used to reduce the temperature so that the function $Temp(t) = \gamma.Temp(t-1)$, $1 < \gamma < 0$, (γ denotes rate of temperature reduction) is assumed. Also, a certain number of iterations (It_0) is considered for the initial temperature to achieve thermal equilibrium. A fully incremental function is used to determine the number of iterations at each temperature; therefore, the intended function $It(t) = It(t-1).\beta$, $1 < \beta < 0$, is considered assuming β as the iteration growth rate at each temperature. Also, the initial main population (*nPop*) is set to *n* and the initial population of neighbors generated by each major member of the population (*nNeighbour*) is set to *m*. We generate the initial population in step 2 (and then in step 3, we estimate the objective value of the initial population one by one. We determine the best solution found from the main population. The condition for stopping the algorithm is to achieve a predetermined number of iterations proportional to the size of the problem

and to reach the end temperature. Steps 7–9 keep the best possible solutions and its objective value up in memory to this phase (if a solution ensures the constraints (10), (11), (12), and (17), then the solution is called a feasible solution. Neighbors are generated for each member of the *SOL* current population according to step 12, and the objective value of the generated neighbors for each i and j is calculated in step 13. In accordance with step 14, i member (in fact, i and j are counters that represent a member of the main population and a member of the *neighbour* population, respectively) is selected from among the best members of the newly created solutions according to the size of the main population. Also, a relative comparison of each member of the main population with one of the selectable neighbors from the new solutions generated is performed according to step 15. According to step 16, if the value of Δ is smaller or equal to zero, then the neighboring solutions win and are accepted, and even if the condition is violated, the neighboring solution is reaccepted with probability $e^{-\frac{\Delta}{Temp}}$ according to step 19 because the end temperature is not set to zero and worse solutions are also accepted. The process of generation of other neighbors by the winning neighbors is continued until the completion conditions of steps 4 and 5 (the maximum number of iteration proportional to the size of the problem and reaching the end temperature) are met. The temperature and the number of iterations at each temperature are updated according to steps 24 and 25, respectively. The coefficients γ and β are set in the range between [0.8, 0.99] and [0.8, 0.99], respectively, to update the temperature value and the maximum number of iterations per temperature. Also, typical number of iterations at each temperature level are in [100, 500]. For example, we initialize the population-based simulated annealing algorithm parameters as follows: $Iter_{max}=100$, $subIter_{max}=10$, $T_0=15$, $T_{end}=4.99$, $\gamma = 0.99$ and $\beta = 0.8$. Also, if we set the number of the main population, n, equal to 5, and the number of the generated neighbors, m, equal to 3, the total number of neighbors generated by the 5 members of the main population will be 15 neighbors. This neighbor generation operation and comparison operation takes place until 15 leads to 4.99. We evaluate 15 members of the generated neighbor population and arrange 15 members of the neighbor population, and thus a competition is created between 15 neighbors. Also, out of the 15 neighbors produced, the best members win according to the number of members of the main population ($n=5$). Each member of the main population is compared to one member of the winning neighbor population according to population-based SA rule (Step 15). First, we update the best solution found. Then, we reduce the temperature according to Step 24 (15*0.99 = 14.85). Next, we update the number of iterations at each temperature according to Step 25 (10/0.8 = 12.5). Finally, we check the condition of the loop ($14.85 \geq 4.99$). This process continues until the final temperature is less than 4.99. In this case, when the final temperature is lower than 4.99, the best locations to place the controllers are selected.

5 Estimation of results

5.1 Experimental settings

The proposed SCLNCPP approach on different topologies (small, medium, large, and very large topologies) has been estimated using the Internet Topology Zoo [28], and its performance has been compared with CNCP and RCCPP formulations. The various topologies we have used include state/regional networks and continental networks. Topologies are divided into four categories based on their size: the first category: small-scale networks ($N < 20$), the second category: medium-scale networks ($20 \leq N < 50$), the third category: large-scale networks ($50 \leq N < 100$), and the fourth category: very large-scale networks ($N \geq 100$). Despite the choice of different types of topologies (e.g., mesh, linear, circular, and hub-and-spoke), we use the latest available version of the topology from the available versions. Topologies used include Sago (18 nodes) from the first category, palmetto (45 nodes) from the second category, uninett (74 nodes) from the third category, and deltacom (112 nodes) from the fourth category.

Latitude and longitude information of the topologies used are available so that we can estimate the delay between nodes. We also use the Floyd-warshall algorithm to estimate the length of the shortest path between nodes. Assuming a maximum access bandwidth of 10Gbps for the server on which the controller software is running, we consider the heterogeneous traffic load generated by the switches to be between 100 and 400 kilo request per second in accordance with [29]. The capacity of each controller is set at 6000 kilo request per second. The size of the intended array is also considered according to the selected network topology. The MIP code and the proposed algorithms are written in MATLAB software R2019b version [30], and we use MATLAB optimization engine and CPLEX IBM ILOG optimizer [31] to obtain the optimal solutions. Also, the system on which all the simulations are performed is as follows: Intel® Core™ i5-4200U CPU @2.30GH and 24 GB RAM with Windows 10 64-bit Enterprise installed on the hard disk.

5.2 Performance metrics

In this paper, we define two performance metrics for comparing the proposed SCLNCPP approach with the CNCP and RCCPP approaches in controller failure case and failure free case, with or without constraints. These two metrics are summarized as worst-case latency in Formula (22) and maximum worst-case latency in Formula (23). Maximum, for all forwarding devices, in other words, the latency from the forwarding device to its closest controller is called the worst-case latency which is mentioned in (22). The maximum latency among the worst-case latencies, for all the different failures, is called the maximum worst-case latency, given in (23).

$$WCL = \max_{v \in N} \left\{ \min_{u \in M} l_{[v][u]} \right\} \tag{22}$$

$$MWCL^S = \max_{s \in S} \left\{ \max_{v \in N} \left\{ \min_{\overline{u} \in M} l^s_{[v][\overline{u}]} \right\} \right\} \tag{23}$$

Here, S is a set of all the different failure scenarios and $l^s_{[v][\overline{u}]}$ is delay from switch v to controller $\overline{u}$ under a failure scenario s. Definition of $l^s_{[v][\overline{u}]}$ in the proposed SCLNCPP approach differs from that of CNCP and RCCPP in (24), (25), and (26), respectively.

$$l^s_{[v][\overline{u}]} = l_{[v][u]} + l_{[v][\overline{u}]} \tag{24}$$

$$l^s_{[v][\overline{u}]} = l_{[v][u]} + l_{[u][\overline{u}]} \tag{25}$$

$$l^s_{[v][\overline{u}]} = l_{[v][u]} + l_{[v][\overline{u}]} \tag{26}$$

In the CNCP approach, in case of controller failures, $l^s_{[v][\overline{u}]}$ is obtained as the sum of the latency from forwarding device v to the first nearest controller (i.e., u) and the latency from the first nearest controller u to the second nearest controller (i.e., $\overline{u}$), while in the RCCPP approach, the objective function is different from the CNCP and corresponds to minimizing the number of controllers. By changing the objective function, the metric $l^s_{[v][\overline{u}]}$ in RCCPP is achieved according to the mathematical formulation in star form from the sum of the latency from forwarding device v to the first controller (not necessarily its closest controller) and the latency from the same switch v to the second controller (not necessarily its closest controller). In the proposed approach, the defined metric is similar to the RCCPP metric, except that we assign the switch v to the first and second nearest controllers in star form to minimize latency. In other words, the metric $l^s_{[v][\overline{u}]}$ in SCLNCPP corresponds to the sum of the latency from forwarding device v to its first closest controller and the latency from the same forwarding device v to its second closest controller.

5.3 Estimation of performance metrics in proposed approach and its comparison with CNCP and RCCPP approaches

The comparison of the worst-case latency and maximum worst-case latency metrics of the proposed SCLNCPP approach with that of the CCNP and RCCPP approaches, taking into account the failure free and one failure scenarios in network, is shown in Fig. 4. Based on the evaluations performed on different topologies, we have experimentally concluded that on average one controller can manage 28 switches; thus, for topologies with 28 nodes or less in all three approaches, CCNP, RCCPP, and SCLNCPP, in failure free and one failure scenarios, at least 2 controllers are enough; but for topologies with more than 28 nodes, two controllers do not seem enough to manage network switches. Even in the evaluations performed, in topologies with more than 100 nodes, even three controllers are not enough; therefore, we

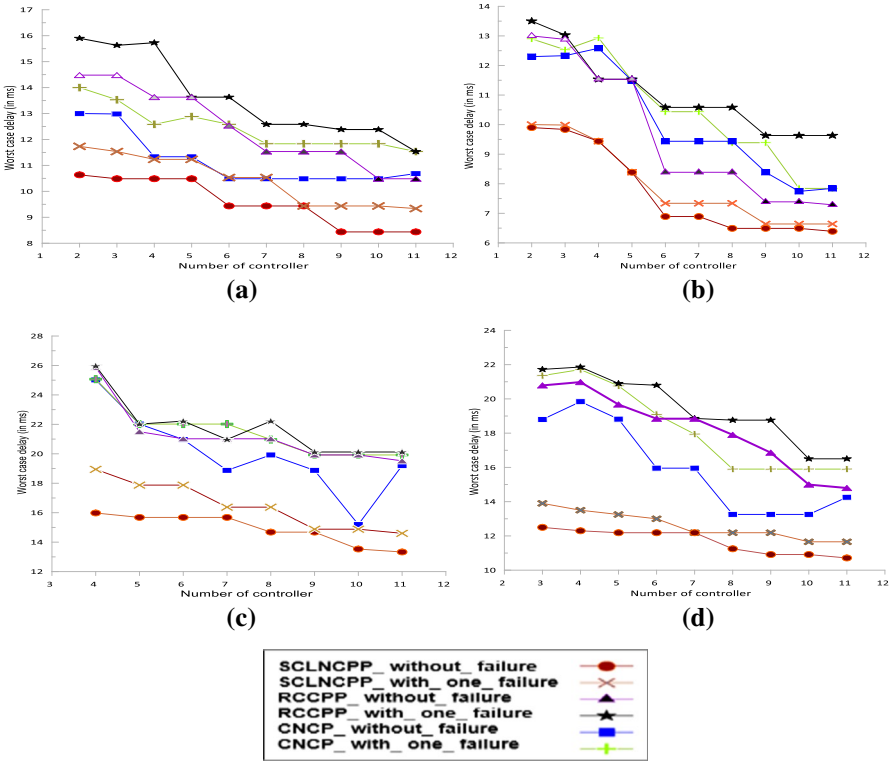

Fig. 4 Worst-case latency and maximum worst-case latency of SCLNCPP, RCCPP and CNCP on various network topologies from small, medium and large scales **a** Sago (18 nodes), **b** Pameltto (45 nodes), **c** Uninett (74 nodes), and **d** Deltacom (112 nodes)

consider the number of controllers for the sago and pameltto topologies to be varied from two to 11. The values obtained from the simulation are shown in Fig. 4a and b. According to estimates, at least three controllers are enough for the CNCP, RCCPP and SCLNCPP approaches when the number of network nodes is between 28 and 84 nodes; therefore, the estimates made for the evaluation of the worst-case latency and the maximum worst-case latency metrics on the uninett topology are presented in Fig. 4c with the number of controllers varying from 3 to 11; however, according to the results obtained, the SCLNCPP approach for topologies with more than 84 nodes with 3 controllers cannot manage the switches in the network. For this reason, we considered the number of controllers for deltacom topology to be varied from 4 to 11, and the results are presented in Fig. 4d.

In Formula (23), we define a metric called worst-case latency to compare the performance of the SCLNCPP, CCNP and RCCPP approaches, which is actually intended for failure free case of the network. As shown in Fig. 4, if there is no failure in the network, the proposed SCLNCPP approach leads to the less worst-case latency compared to both the CCNP and RCCPP approaches, because SCLNCPP with its array approach has significantly reduced the network search

space (in other words, controllers in the network are placed in a compact $\frac{NC}{2}$ bit search space, instead of being placed in $2NC$ bits or NC bit space). Also, CNCP is better than RCCPP in terms of the above metric because in RCCPP, due to its formulation, there is no need to assign the switches of the failed controllers of the network to the second nearest controller, which increases the worst-case latency shown in Fig. 4b, c, and d. Only in Fig. 4a at $N_c = 6$, RCCPP performed better than CNCP because of the type of the Sago topology, which is a linear topology with closely related nodes.

In Formula (23), we defined the maximum worst-case latency metric to compare the said approaches in the case of different controller failure scenarios. According to Fig. 4, it can be seen that SCLNCPP performs better than CCNP and RCCPP because in case of controller failures, in addition to reducing the network search space, the type of the planning of assignment of switches to the controllers is star, and for reassignment, it considers the second nearest controller to the switch itself in case of failure. Although the CNCP also considers the second nearest controller (the second nearest controller is the controller closest to the first controller) for reassignment in case of failure, the type of the planning is linear.

Maximum worst-case latency of CNCP increases with the number of controllers in the network when we have controller failure in the network, because their purpose is to minimize the worst-case latency. This is also true for the SCLNCPP approach, but the worst-case latency of CNCP, in failure free case, violates this process, as shown in Fig. 4c and d; therefore, it can be concluded that the placement of the controllers in the locations where the maximum worst-case latency metric is minimized does not guarantee that the worst-case latency decrease as the number of controllers increases, because the way the switches are assigned to the controllers is linear. In the star assignment models of SCLNCPP and RCCPP, the above cases are violated, that is, minimizing the maximum worst-case latency metric ensures minimizing the worst-case latency due to an increase in the number of controllers in failure free case.

In order to ensure the proper operation and maintain the consistent global view of the distributed SD-WAN, the controllers deployed in the network must communicate with each other. In fact, the latency between the farthest controllers in the network is called the worst-case inter-controller latency. In the following, the results of the comparison of the simulations of this metric are shown in Fig. 5. In the CNCP approach, the purpose is to minimize the worst-case latency between the forwarding devices and their second controllers (the second controller assigned to the forwarding device is the controller closest to the first controller of the forwarding device), if the first nearest reference controller of switches is failed. Because of this, the CNCP does not force the first and second controllers of a forwarding device to be placed near to each other. In RCCPP, first, the switch is assigned to the first and second controllers in star form by the SMC-RCCPP and AMC-RCCPP algorithms; second, it is not necessary to assign the switch to its second nearest controller, but by using a constraint on the cc_{max} parameter, the switch is randomly assigned to the first and

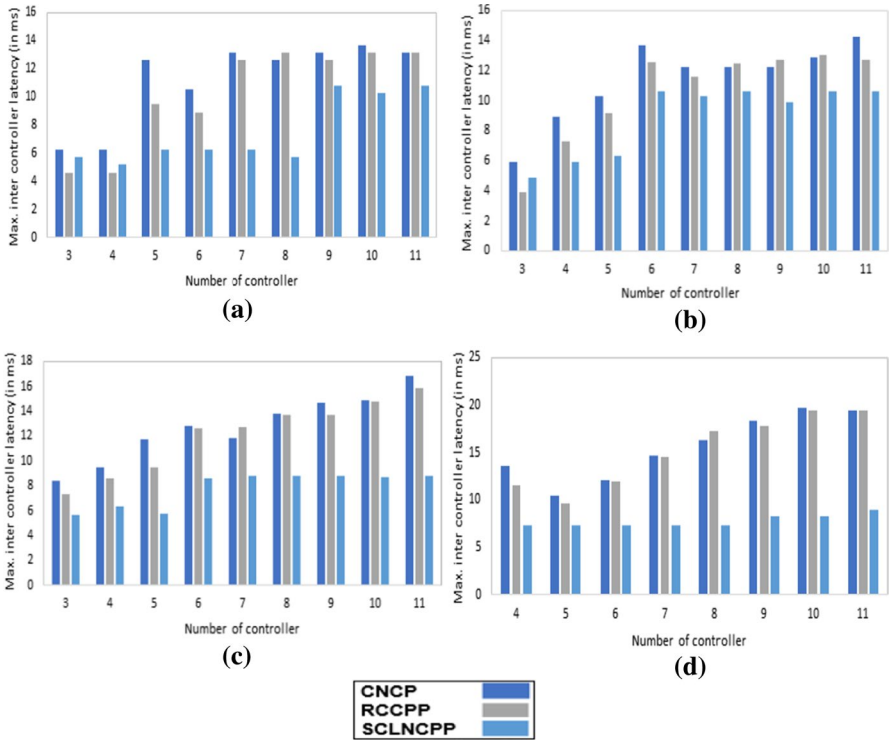

Fig. 5 Maximum inter-controller latency of SCLNCPP, RCCPP and CNCP on various network topologies from small, medium and large scales **a** Sago (18 nodes), **b** Pameltto (45 nodes), **c** Uninett (74 nodes), and **d** Deltacom (112 nodes)

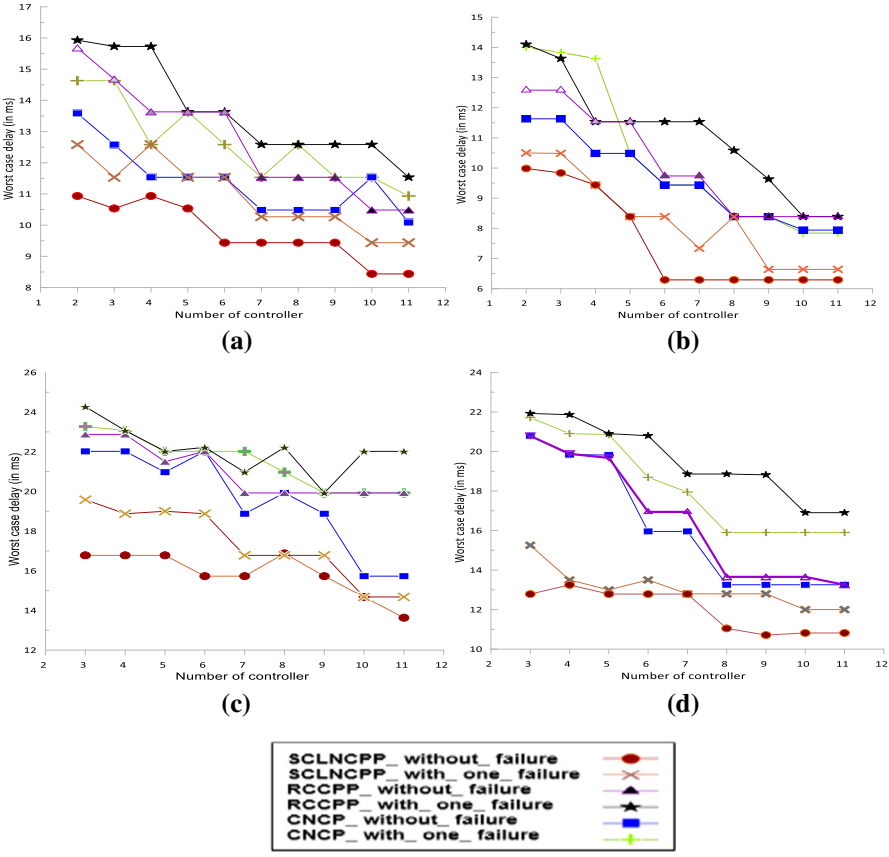

Fig. 6 Worst-case latency and maximum worst-case latency of SCLNCPP, RCCPP and CNCP on various network topologies from small, medium and large scales, despite limiting the maximum latency between controllers **a** Sago (18 nodes), **b** Pameltto (45 nodes), **c** Uninett (74 nodes), and **d** Deltacom (112 nodes)

second controllers. Also, in RCCPP, the controllers of different forwarding devices are not forced to be placed close to each other.

In the proposed SCLNCPP approach, the approach of assignment of the switch to the first and second controllers is performed in star form, but in this case, if the first controller of the switch is failed, the switch is assigned to its second nearest controller (not to the second nearest controller of the first controller linearly). Therefore, in CNCP and RCCPP approaches, the inter-controller latency increases with the number of controllers, but in SCLNCPP, the same increasing process is followed in the inter-controller latency, but it is not significant. Also, the inter-controller latency values of SCLNCPP are much lower than that of the CNCP and RCCPP because this approach, with its array formulation, forces the switches of the different controllers to be close together. As is seen in Fig. 5, the maximum inter-controller latency of SCLNCPP is much less than that of the CNCP and RCCPP. Also, to compare CNCP and RCCPP, it can be said that the maximum

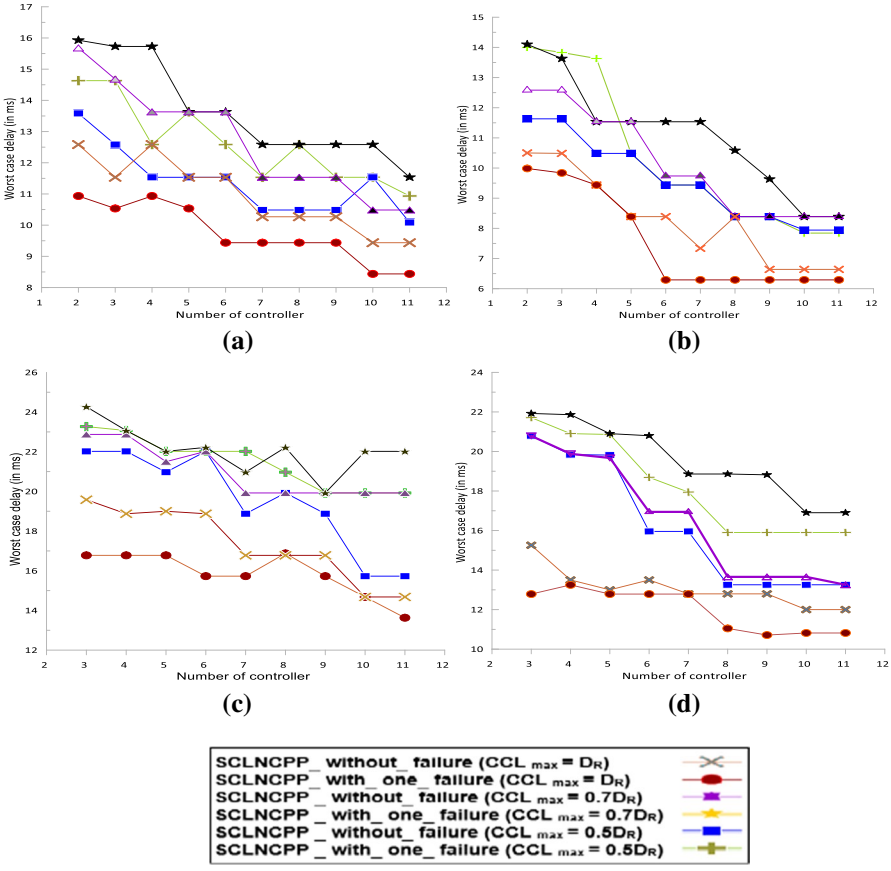

Fig. 7 Worst-case latency and maximum worst-case latency of the proposed SCLNCPP approach on various network topologies from small, medium and large scales, despite limiting the maximum latency between controllers as a factor of array diameter **a** Sago (18 nodes), **b** Pameltto (45 nodes), **c** Uninett (74 nodes), and **d** Deltacom (112 nodes)

inter-controller latency of RCCPP is better than that of CNCP when fewer controllers are placed in the network and it is near to CNCP when more than the required number of controllers are installed in the network; this is because in the RCCPP approach, it is not necessary to assign the switch to the second nearest controller, and because the switch-controller latency and inter-controller latency are two competitive metrics, when the switch-controller latency increases, the inter-controller latency decreases. By placing more than the required number of controllers in the network, the cost of the network increases; therefore, in order to reduce the cost of the network, the network designer prevents the deployment of more than the required number of controllers.

To solve the problem of placing controllers more than the required number, we have to limit the maximum inter-controller latency using constraints (13) to (16). Next, we consider the effect of limiting the maximum worst-case inter-controller

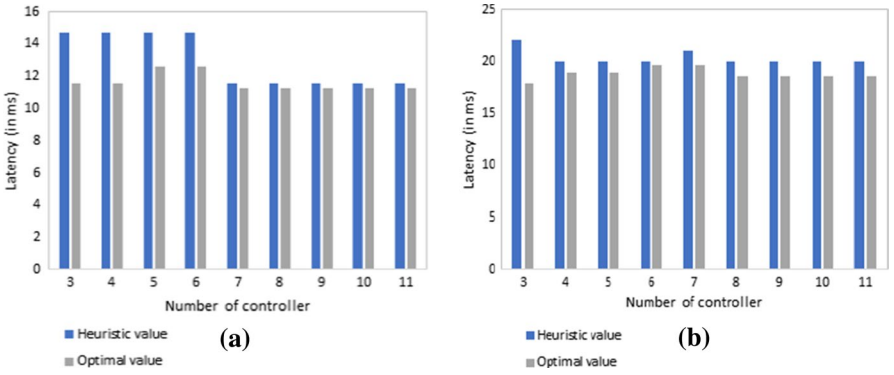

Fig. 8 Performance of the population-based simulated annealing algorithm in terms of objective value or best cost on topologies **a** Pameltto (45 nodes) and **b** Deltacom (112 nodes)

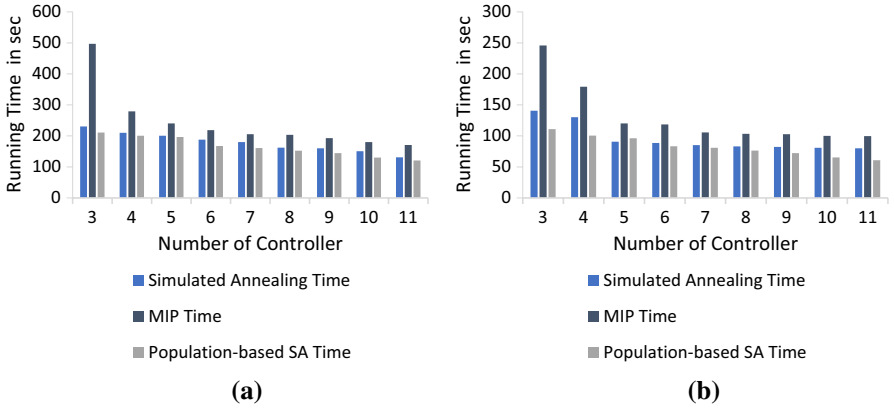

Fig. 9 Performance of the population-based simulated annealing algorithm in terms of running time on topologies **a** Pameltto (45 nodes) and **b** Deltacom (112 nodes)

latency as a coefficient of array diameter (i.e., $CCL_{max} = 0.6D_R$) on the worst-case latency and maximum worst-case latency of the SCLNCPP, RCCPP and CNCP approaches between the switches and second controllers in Fig. 6. Figure 6 shows that SCLNCPP performs better than CNCP and RCCPP in case of controller failures due to the reasons stated above. In comparing CNCP with RCCPP, it can be concluded that CNCP performs better than RCCPP in case of controller failures, and in some points the values are close to each other, as shown in Fig. 6b and d. In failure free case, SCLNCPP operates better than both the CNCP and RCCPP approaches, but in comparing CNCP with RCCPP, in failure free case, it can be seen that CNCP is better than RCCPP, and in some points the values are close to each other, which is evident in Fig. 6c and d.

The results of the comparison of the worst-case latency and maximum worst-case latency metrics of the proposed SCLNCPP approach are shown in Fig. 7 when the

maximum latency between controllers is limited to some coefficients of the network diameter. We see a sudden surge in the maximum worst-case latency of SCLNCPP when we limit the maximum latency between controllers. According to Fig. 6 on various topologies, it can be concluded that the values of the worst-case latency and the maximum worst-case latency of SCLNCPP, CNCP, and RCCPP with the latency inter-controller limits are higher compared to that of SCLNCPP, CNCP, and RCCPP without these limits. The reasons for the sudden surge observed in different topologies also depend on the extent to which the latency between the controllers is limited as shown in Fig. 5.

5.4 Quality of our heuristic algorithm solution and its comparison with optimal value

The population-based simulated annealing heuristic algorithm is implemented in MATLAB software and its performance is compared to MIP in terms of latency or objective value and running time. The starting and end temperatures are 15 and 4.99, respectively. The number of iterations in each temperature level, $Iter_{max}$, is equal to 100, the temperature reduction coefficient (γ) is equal to 0.99 and β as the iteration growth rate at each temperature is equal to 0.8. Each sample is run for 100 times and the average results are presented per run for statistical reliability. Figure 8 shows

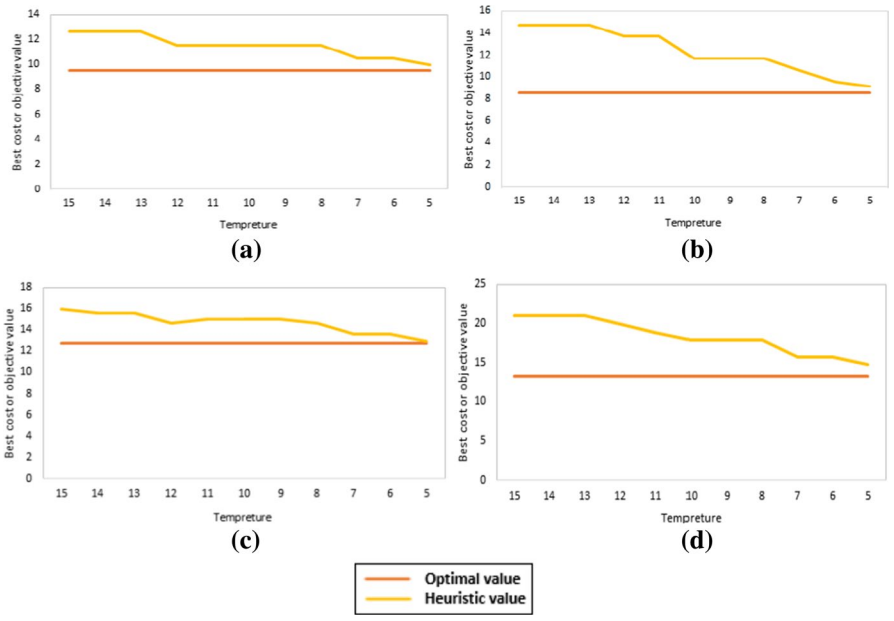

Fig. 10 Behavior of a population-based simulated annealing algorithm in terms of convergence to the optimal value on topologies **a** Pameltto (45 nodes) with 3 controllers, **b** Pameltto (45 nodes) with 4 controllers, **c** Deltacom (112 nodes) with 3 controllers, and **d** Deltacom (112 nodes) with 4 controllers

the performance of the population-based simulated annealing algorithm on pameltto and deltacom topologies for different values of the number of controllers in Fig. 8a and b, respectively. According to the figure, we observe that there is a very small gap between the optimal value and the amount of output generated by the heuristic algorithm unless this gap may be further reduced by increasing γ and $Iter_{max}$ and decreasing β. Also, Fig. 9 illustrates the execution time of the MIP by CPLEX, the simulated annealing heuristic and the population-based simulated annealing heuristic on pameltto and deltacom topologies. We can observe that the population-based simulated annealing heuristic algorithm takes less than the half of the time taken by MIP. In addition, we can show that the population-based simulated annealing heuristic algorithm performs better than simulated annealing heuristic in term of running time.

The behavior of the population-based simulated annealing heuristic algorithm on the pameltto topology for the $N_c = 3$ and $N_c = 4$ controllers is shown in Fig. 10a and b, respectively. It can also be seen on the deltacom topology for $N_c = 3$ and $N_c = 4$ in Fig. 10c and d. The population-based simulated annealing algorithm, by accepting worse solutions with high probability at higher temperatures, leads to a gap between the objective value of the heuristic algorithm and the optimal value at higher temperatures. As the temperature decreases, the probability of accepting worse solutions decreases and the algorithm converges to the optimal value. Also, whatever the size of the intended topology is larger, the gap between the target (objective) value of the heuristic algorithm and the optimal value is smaller because the algorithm explores different parts of the network search space using population concept (not being single point) and converges toward the optimal value by not getting caught up in local optimum.

5.5 Comparison of proposed SCLNCPP approach with CNCP and RCCPP approaches

In the final part of the paper, we compare the proposed SCLNCPP approach with the two CNCP and RCCPP approaches from different aspects (such as formulation type, objective delay, load imbalance, approach application type, etc.). To compare SCLNCPP with CNCP and RCCPP approaches, we also need to calculate the optimal solutions. The most obvious difference between our proposed mathematical formulation (i.e., SCLNCPP) and the two CNCP and RCCPP approaches is the use of array decision variables. In other words, in the proposed formulation, we used the decision variables $AP^2_{[v][u]}$ and $AP^1_{[v][u]}$, respectively, to assign the forwarding device to the first closest controller and assign the forwarding device to the second closest controller in case of one controller failure. In contrast, in the CNCP approach, considering one controller failure, the two-indexed decision variable r^1_{ij} and r^2_{ij} is used, and in mathematical formulation by extending the failure to multiple controllers, the three-indexed decision variable r^l_{ijk} is used. In addition, RCCPP formulation, a two-indexed decision variable x_{ij} is used to assign a switch to multiple controllers. Another obvious difference is the change of the mathematical formulation of SCLNCPP from linear form to star form, which performs the operation of assigning static switches to controllers in star form. Also, we consider the maximum inter-controller latency (i.e.,$CCLB_{max}$) as a coefficient of the array diameter (corresponding to a factor of the graph diameter in RCCPP

Table 4 The rate of improvement of the SCLNCPP approach compared to the RCCPP and CNCP approaches in terms of latency metrics

Topology	Network size	cc_{max}	SCLNCPP improvement rate compared to CNCP in case of failure	SCLNCPP improvement rate compared to CNCP in failure free case	SCLNCPP improvement rate compared to RCCPP in case of failure	SCLNCPP improvement rate compared to RCCPP in failure free case
Sago	18	$0.6D_R$	2.19 ms	2.081 ms	3.08 ms	2.66 ms
Pameltto	45	$0.6D_R$	1.57 ms	1.7 ms	2.785 ms	2.95 ms
Uninett	74	$0.6D_R$	5.25 ms	4.3 ms	6.54 ms	4.53 ms
Deltacom	112	$0.6D_R$	4.11 ms	3.58 ms	4.76 ms	5 ms
Sago	18	$0.7D_R$	2.25 ms	2.2 ms	3.2 ms	2.8 ms
Pameltto	45	$0.7D_R$	1.6 ms	1.9 ms	2.858 ms	3 ms
Uninett	74	$0.7D_R$	5.4 ms	4.4 ms	6.7 ms	4.7 ms
Deltacom	112	$0.7D_R$	4.3 ms	4.1 ms	4.85 ms	5.2 ms
Sago	18	$0.5D_R$	1.9 ms	1.95 ms	2.9 ms	2.5 ms
Pameltto	45	$0.5D_R$	1.3 ms	1.45 ms	2.6 ms	2.8 ms
Uninett	74	$0.5D_R$	4.7 ms	4 ms	5.89 ms	4.23 ms
Deltacom	112	$0.5D_R$	5 ms	4.1 ms	4.6 ms	4.8 ms

(similar to the SCLNCPP approach)) and corresponding to γ in the CNCP (difference with the SCLNCPP approach). Also, we consider the number of controllers placed in the intended topology equal to N_c (corresponding to parameter Q in CNCP and corresponding to r in RCCPP), controller capacity equal to $U_{[u]}$ (corresponding to u_c in RCCPP and U_j in CNCP) and $Load_{[v]}$ in the form of a homogeneous traffic load. Finally, in the analysis of the results in the study of Killi and Rao [8], the estimates are not performed on the worst-case latency and maximum worst-case latency metrics for topologies with more than 40 nodes. Also, optimal results are not obtained in a reasonable time by Gurobi solver in RCCPP for topologies with more than 80 nodes, but in the proposed approach, estimates for topologies with more than 80 nodes respond with either the heuristic approach or the CPLEX solver.

6 Conclusions and future orientations

We presented a Star Capacity-aware Latency-based Next Controller Placement Problem (called SCLNCPP) in the SD-WAN, which actually prevents a lack of connections and a dramatic increase in the worst-case delay in case of controller failures. The purpose of this approach is to minimize the maximum, for all forwarding devices, of the sum of the latency from the forwarding device to the first nearest controller and the latency from the same switch to the second nearest controller. We provided the proposed approach formulation as a MIP with array decision variables for a controller failure. We also extended it to multiple controller failures. Also, we even present a population-based simulated annealing heuristic algorithm to solve the problem optimally. In addition, evaluations of the various Internet Zoo topologies were estimated, and the result is that the proposed SCLNCPP approach performs better than CNCP and RCCPP in both case of failure and failure free case. Table 4 shows the rate of improvement of the SCLNCPP approach in terms of latency metric compared to the two approaches of CNCP and RCCPP in case of failure and free failure case with Considering 11 controllers in the network topology. According to Table 4, it is evident that the rate of latency in large-scale topologies for the proposed approach compared to CNCP and RRCCPP approaches is improved. The results show that the heuristic algorithm converges faster to an optimal value in large-scale topologies due to not being single point and exploring different parts of the network search space using the population concept. In future work, we will use partitioning-based heuristic algorithms such as BBO and EADUC to solve the problem optimally. In our orientation, we will also look at the dynamics of the SCLNCPP approach to change switch-controller assignments based on variable traffic loads over time. Finally, we will try to use this approach to reduce the network search space on hybrid networks (such as IOT/SDN or 5G/SDN), and even we will apply the mobility on mobile networks that require network dynamics control.

Acknowledgments The authors would like to thank Dr. Killi Prakasa for his advice on the field of CPP in SDNs.

References

1. Shirmarz A, Ghaffari A (2021) Taxonomy of controller placement problem (CPP) optimization in Software Defined Network (SDN): a survey. J Ambient Intell Humaniz Comput 12:10473–10498. https://doi.org/10.1007/s12652-020-02754-w
2. Rehman AU, Rui L, Aguiar BJP (2019) network functions virtualization: the long road to commercial deployments. Access IEEE 7:60439–60464
3. Tanha M, Sajjadi D, Ruby R, Pan J (2018) Capacity-aware and delay-guaranteed resilient controller placement for software-defined WANs. IEEE Trans Net Serv Manag 15(3):991
4. Heller B, Sherwood R, McKeown N (2012) The controller placement problem. First workshop on Hot topics in software defined networks, Rev 42(4):7–12
5. Killi BPR, Rao SV (2019) Towards improving resilience of controller placement with minimum backup capacity in software defined networks. Comput Netw 149:102–114
6. Killi BPR, Rao SV (2016) Optimal model for failure foresight capacitated controller placement in software defined networks. Proc IEEE Communication Letter 20(6):1108–1111
7. Killi BPR, Rao SV (2017) Capacitated next controller placement in software defined networks. IEEE Trans Netw Serv Manage 14(3):514–527
8. Sallahi A, St-Hilaire M (2016) Optimal model for the controller placement problem in software defined networks. IEEE Communication Letter 19(1):30–33
9. Zhang T, Giaccone P, Bianco A, Domenico SD (2017) The role of the inter-controller consensus in the placement of distributed SDN controllers. Computer Communication
10. Muller LF, Oliveira RR, Luizelli MC, Gaspary LP, Barcellos MP (2014) Survivor: an enhanced controller placement strategy for improving SDN survivability. In: Proc. IEEE GLOBECOM, pp 1909–1915
11. Hock D, Gebert S, Hartmann M, Zinner T, Tran-Gia P (2014) POCO-framework for Pareto-optimal resilient controller placement in SDN-based core networks. In: Proc. IEEE/IFIP NOMS, pp 1–2
12. Jimenez Y, Cervello-Pastor C, Garsia AJ (2014) On the controller placement for designing a distributed SDN control layer. In: Proc. IFIP NETWORKING, pp 1–9
13. Wang G, Zhao Y, Huang J, Wang W (2018) The controller placement in software defined networking: a survey. Proc IEEE network 31(5):21–27
14. Fan Y, Xia Y, Liang W, Zhang X (2017) Latency-aware reliable controller placements in SDNs. In: Proc. ChinaCom/ICCN 210:152–162
15. Killi BPR, Rao SV (2018) Link failure aware capacitated controller placement in software defined networks. In: Proc. IEEE/ICOIN
16. Killi BPR, Rao SV (2020) Poly-stable matching based scalable controller placement with balancing constraints in SDN. Comput Commun 154:82–91
17. Kanodia K, Mohanty S, Sahoo B, Kurroliya K (2020) HPSOSA: A hybrid approach in resilient controller placement in SDN. In: 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, pp 1–6. https://doi.org/10.1109/ic-ETITE47903.2020.467
18. Kanodia K, Mohanty S, Kurroliya K, Sahoo B (2020) CCPGWO: A meta-heuristic strategy for link failure aware placement of controller in SDN. In: 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, pp. 859–863. https://doi.org/10.1109/ICICT48043.2020.9112423
19. Khuller S, Sussmann YJ (2000) The capacitated k-center problem. Proc SIAM J Discrete Math 13(3):403–418
20. Kuang H, Qiu Y, Li R, Liu X (2018) A hierarchical K-means Algorithm for controller placement in SDN-based WAN architecture. In: IEEE 10th International Conference on Measuring Technology and Mechatronics Automation
21. Behsaz B, Salavatipour MR, Svitkina Z (2016) New approximation algorithms for the unsplittable capacitated facility location problem. Algorithmica 75(1):53–83
22. Khuller S, Sussmann YJ (2000) The capacitated k-center problem. SIAM J Discrete Math 13:403–418
23. ONF (2015) OpenFlow switch specification-version 1.5.1. https://goo.gl/jE2JTW
24. Obadia M, Bouet M, Leguay J, Phemius K, Iannone L (2014) Failover mechanisms for distributed sdn. Conference and Workshop on the Network of the Future (NOF), Dec controllers, in Proc. International, pp 1–6

25. Lange S, Gebert S, Zinner T, Tran-Gia P, Hock D, Jarschel M, Hoffmann M (2015) Heuristic approaches to the controller placement problem in large scale SDN networks. IEEE Trans Netw Serv Manage 12(1):4–17
26. Wagner JL, Falkson LM (1975) The optimal nodal location of public facilities with price-sensitive demand. Geogr Anal 7(1):69–83
27. Askarzadeh A, Coelho L. S, Klein C. E, Mariani V. C (2016) A population-based simulated annealing algorithm for global optimization. In: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC). https://doi.org/10.1109/SMC.2016.7844961
28. Knight S, Nguyen HX, Falkner N, Bowden R, Roughan M (2011) The internet topology zoo. IEEE J Sel Areas Commun 29:1765–1775
29. Yao G, Bi J, Li Y, Guo L (2014) On the capacitated controller placement problem in software defined networks. IEEE Communication Letter 18(8):1339–1342
30. MATLAB version 8.5.0.197613 (R2019b) (2019) The Mathworks, Inc., Natick, Massachusetts
31. IBM ILOG CPLEX. [Online]. Available: http://www-01.ibm.com/software/integration/optimization/cplex- optimizer

Authors and Affiliations

Hadi Mojez[1] · Amir Massoud Bidgoli[1] · Hamid Haj Seyyed Javadi[2]

✉ Amir Massoud Bidgoli
am_bidgoli@iau-tnb.ac.ir

Hadi Mojez
h.mojez@iau-tnb.ac.ir

Hamid Haj Seyyed Javadi
h.s.javadi@shahed.ac.ir

[1] Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran

[2] Department of Mathematics and Computer Sciences, Shahed University, Tehran, Iran