Controller Placements for Optimizing Switch-to-Controller and Inter-controller Communication Latency in Software Defined Networks

Bo Wei, Yuqi Fan(✉), Lunfei Wang, Tao Ouyang, and Lei Shi

School of Computer and Information Engineering, Anhui Province Key Laboratory of Industry Safety and Emergency Technology, Hefei University of Technology, Hefei 230601, Anhui, China
{2019110966,cxwlf,ouyangtao}@mail.hfut.edu.cn,
{yuqi.fan,shilei}@hfut.edu.cn

Abstract. The logically centralized control plane in large-scale SDN networks consists of multiple controllers, and the controllers communicate with each other to keep a consistent view of the network status. Inconsistent controllers or controllers failing to maintain the network state in time may severely degrade the network performance. However, most of the existing research focuses on the switch-to-controller communication latency by ignoring the communication latency between the controllers. In this paper, we formulate a novel multi-objective SDN controller placement problem with the objectives to minimize both the switch-to-controller and the inter-controller communication latency. We propose an efficient Multi-Objective Controller Placement (MOCP) algorithm. Algorithm MOCP generates the new controller placement solutions with crossover and mutation operations. The switches are assigned to the controllers with the greedy strategy initially, and then mapped to the other controllers via the local search strategy. Our simulation results show that algorithm MOCP can effectively reduce the latency between the controllers and from the switches to controllers simultaneously.

Keywords: Software defined networks · Latency · Multi-objective optimization · Non-dominated solution

1 Introduction

In Software Defined Networks (SDNs), network switches (nodes) are only responsible for data forwarding, while controllers determine the paths of network packets across the switches. Upon the arrival of an unknown flow, the switch sends

L. Shi—This work was partly supported by the National Key Research and Development Plan (2018YFB2000505), the Key Research and Development Project in Anhui Province (201904a06020024), and the Open Project of State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System in China (CEMEE2018Z0102B).

Z. Liu et al. (Eds.): WASA 2021, LNCS 12937, pp. 535–546, 2021.
https://doi.org/10.1007/978-3-030-85928-2_42

a flow set-up request to the controller which responds to the request with a flow entry to be installed in the flow table of the switch. Therefore, the switch-to-controller latency is critical for the performance of SDNs.

Some work has been done to improve the latency performance in SDNs, which is important in computer networks [1–3]. The network was split into several sub-networks using community partitioning and a controller was deployed in each sub-network [4]. A capacitated controller placement problem was introduced to minimize the propagation delay, whereas the load of each controller does not exceed its capacity [5]. The controller placement problem in an edge network was formulated with the objectives of delay and overhead minimization; the problem was converted to a Mixed Integer Programming (MIP) problem using linearization techniques, and approximation solutions were presented using the theory of supermodular functions [6]. A Pareto-based optimal controller placement method (POCO) was proposed to consider maximum node-to-controller latencies and resilience in terms of failure tolerance and load balancing [7]. The POCO framework was extended with heuristics to support large-scale networks or dynamic networks with the properties changing over time [8]. The controller placement problem was investigated by jointly taking into account both the communication reliability and the communication latency between controllers and switches if any link in the network fails [9]. A metaheuristic-based Reliability-Aware and Latency-Oriented controller placement algorithm (RALO) was proposed to minimize the switch-to-controller communication delay for both the cases without link failure and with single-link-failure [10].

Inter-controller and controller-node traffic overheads can be at the same order of magnitude [6], and existing research indicates that inconsistent controllers or the controllers that fail to maintain the state of the network in time, will not only affect network performance, but also severely degrade the performance of some application layer applications [11]. Existing research on communication latency oriented controller placements focuses on how to reduce the switch-to-controller latency, without considering the communication latency between the controllers.

In this paper, we address the controller placement problem to reduce the communication latency from the switches to the controllers and between the controllers. We formulate a novel multi-objective SDN controller placement problem with the aim to minimize both the average switch-to-controller delay and the maximum inter-controller communication latency. We propose an efficient metaheuristic-based Multi-Objective optimization Controller Placement algorithm (MOCP) for the problem. Algorithm MOCP generates the new controller placement solutions with crossover and mutation operations. The switches are assigned to the controllers with the greedy strategy initially, and then mapped to the other controllers via the local search strategy. Finally, we conduct experiments through simulations. Experimental results demonstrate that algorithm MOCP can effectively reduce the latency between the controllers and from the switches to controllers simultaneously.

2 Problem Formulation

Table 1. Table of symbols and notations

s_i, s_j	i-th, j-th switch/node
c_k	k-th controller
C	The set of controllers
K	The number of controllers
u_k	The capacity of controller c_k
r_i	The number of requests from switch s_i
$x_{i,k}$	Indicate whether switch s_i is associated with controller c_k $(= 1)$ or not $(= 0)$
$y_{i,k}$	Indicate whether controller c_k is co-located with switch s_i $(= 1)$ or not $(= 0)$
$l_{i,j}$	The shortest path latency between nodes s_i and s_j

For a given SDN network $G = (V, E)$, where V is the set of switches/nodes and E denotes the set of edges between the switches. Each controller is co-located with one and only one switch [7], and the total number of requests processed by each controller should be within its processing capacity. Each switch is mapped to exactly one controller. When a switch is mapped to a controller, we say the switch and the controller are associated with each other. The symbols and notations used in the paper are listed in Table 1.

The communication between two nodes goes through the shortest path between the two nodes. In this paper, we aim to determine where to place each controller and the exact association relationship between the controllers and the switches, with the objectives to optimize both the average switch-to-controller delay and the maximum inter-controller communication latency. In other words, our optimization objectives are to

Minimize:

$$[l^c, l^s] \tag{1}$$

Subject to:

$$l^c = \max_{c_k \in C, c_{k'} \in C} d_{k,k'} \tag{2}$$

$$l^s = \frac{\sum_{i=1}^{|V|} \sum_{k=1}^{K} l_{i,k} \cdot x_{i,k}}{|V|} \tag{3}$$

$$\sum_{k=1}^{K} x_{i,k} = 1, \quad \forall s_i \in V \tag{4}$$

$$\sum_{i=1}^{|V|} y_{i,k} = 1, \quad \forall c_k \in C \tag{5}$$

$$y_{i,k} \leq x_{i,k}, \quad \forall c_k \in C,\ \forall s_i \in V \tag{6}$$

$$\sum_{i=1}^{|V|} x_{i,k} \cdot r_i \leq u_k, \quad \forall c_k \in C \tag{7}$$

$$x_{i,k} \in \{0,1\}, y_{i,k} \in \{0,1\}, \quad \forall c_k \in C \tag{8}$$

Equations (2) and (3) define the maximum latency between controllers and the average latency from the switches to the associated controllers, respectively. Equation (4) ensures that each switch is mapped to one and only one controller. Equation (5) mandates that each controller can only be co-located with one switch. Equation (6) dictates that switch s_i must be mapped to controller c_k, if controller c_k is co-located with switch s_i. Equation (7) signifies that the controllers cannot be overloaded. Equation (8) requires that $x_{i,k}$ and $y_{i,k}$ are binary integer variables.

3 Controller Placement Algorithm

In this section, we propose an efficient metaheuristic-based Multi-Objective Controller Placement (MOCP) algorithm to obtain a set of non-dominated solutions. Each solution determines the locations of the controllers. Algorithm MOCP shown in Algorithm 1 first randomly generates an initial solution set P_0 with N solutions, and performs crossover and mutation operations to get another solution set Q_0 with N solutions (lines 1–2). The algorithm then proceeds iteratively. In each iteration, the algorithm merges the last obtained solution sets P_w and Q_w to get a new solution set R_w with $2N$ solutions (line 4). After assigning the switches to the controllers for each solution in R_w, the algorithm performs solution ranking and solution evaluation to find the best N solutions in R_w as the new solution set P_{w+1}. A new solution set Q_{w+1} is also generated by performing crossover and mutation operations on P_{w+1} (line 7). The procedure continues until the maximum number of iterations is reached. The algorithm returns O_1 obtained by solution ranking as the Pareto optimal solution set.

3.1 Solution Construction

Assuming P is the existing set of N solutions, the algorithm constructs Q, a new set of N solutions, by performing crossover and mutation operations on the solutions in P.

With crossover, two solutions exchange the controller placement locations between two designated crossover points. Mutation places one of the controllers

Algorithm 1. *MOCP algorithm*

Input:

$G = (V, E)$; K; Solution set size N; Number of iterations I.

Output:

The controller locations and the mapping relationship between the switches and the controllers.

1: Randomly generate the initial population P_0 with N solutions;
2: Perform crossover and mutation on P_0 to obtain a new solution set Q_0, $w = 0$;
3: **while** $w \leq I$ **do**
4: $\quad R_w = P_w \cup Q_w$;
5: $\quad$ Run Algorithm *Switch Assignment* on R_w;
6: $\quad$ Perform *Solution Ranking*, *Congestion Degree Calculation* on R_w, and select the best N solutions among R_w to obtain solution set P_{w+1};
7: $\quad$ Perform crossover and mutation operations on P_{w+1} to get another solution set Q_{w+1} with N solutions;
8: $\quad w = w + 1$;
9: **end while**
10: Return O_1.

in the current solution at another node. Assuming the position of controller c_k to be mutated is node s_i in the current solution, the position of controller c_k is mutated to another node which is adjacent to s_i. If there are multiple such nodes, we randomly select one as the new controller location of c_k.

3.2 Evaluation of the Solutions

For two solutions p and p', we say p dominates p' ($p \prec p'$), if solution p is better than p' in both of the objectives. If there is no solution in solution set P dominates p ($\nexists p' \prec p, p' \in P$), p is called a non-dominated solution. The set of all non-dominated solutions is called the non-dominated solution set or the Pareto optimal solution set.

For an existing solution set P, solution ranking shown in Algorithm 2 ranks the solutions by the ascending order the number of solutions dominate the solutions. The solutions of the same rank are put in the same solution subset. The algorithm first puts the solutions which cannot be dominated by any other solution in P in the same subset O_1 (lines 1–4). For each solution o in O_1, the algorithm finds all the solutions that can be dominated by o, and denotes the collection of the found solutions as Q_o. For each solution $q \in Q_o$, the algorithm deceases the value of e_q by 1. If $e_q = 0$, we put solution q in subset O_2. Obviously, the solutions in O_2 are worse than those in O_1. The algorithm continues the process of putting each solution in P into a subset, until all the solutions are classified in the subsets. We denote $Rank(p) = r$, if solution $p \in O_r$.

Congestion degree of solution p, denoted as $Cong(p)$, is used to estimate the intensity of other solutions around solution p, which can be expressed graphically as the side length of the largest rectangle around solution p. For each of the two objective functions, Algorithm 3 sorts the solutions in solution set P by the

Algorithm 2. *Solution Ranking*

Input:
Solution set P.
Output:
The ranked solution subsets.

for each solution $p \in P$ **do**
  Calculate e_p, the number of the solutions in P dominating solution p;
end for
Put each solution $p \in P$ with $e_p = 0$ in solution subset O_1; $r = 1$;
while $O_r \neq \phi$ **do**
  $r = r + 1$;
  for each solution $o \in O_{r-1}$ **do**
    Find the solution collection Q_o dominated by solution o;
    if $Q_o \neq \phi$ **then**
      for each solution $q \in Q_o$ **do**
        $e_q = e_q - 1$;
        if $e_q = 0$ **then**
          Add solution q into solution subset O_r;
        end if
      end for
    end if
  end for
end while
Return each O_r.

non-ascending order of the objective function values. Denote the two objective function values of solution p as $f_1(p)$ and $f_2(p)$, respectively. Let the congestion degrees of the two solutions with the smallest and largest objective function values be ∞. For each of the other solutions, the congestion degree of solution p is calculated as the total side length of the rectangle around solution p on the two objective functions.

After performing solution ranking and congestion degree calculation, each solution p in current solution set P has two attribute values: solution rank $Rank(p)$ and congestion degree $Cong(p)$. For solutions p and p', we say p is better than p' if one of the two conditions is satisfied: (1) $Rank(p) < Rank(p')$, or (2) $Rank(p) = Rank(p')$ and $Cong(p) > Cong(p')$.

3.3 Switch to Controller Assignment

Algorithm 4 shows the process of allocating the switches to the controllers given the controller locations. Each controller maintains a list b_k of the switches, and the switches in each list are sorted in the non-descending order of the shortest path lengths between the controller and switches (lines 1–4). Algorithm 4 selects the switch $s_{i'}$ with the smallest distance to its corresponding controller from all the header nodes of the K lists (lines 6–12). Algorithm 4 then assigns the selected switch $s_{i'}$ to the controller $c_{k'}$, such that the path length between $s_{i'}$

Algorithm 3. *Congestion Degree Calculation*

Input:
Solution set P.
Output:
The congestion degree of each solution $p \in P$.

```
Initialize each Cong(p) = 0;
for g = 1..2 do
    Sort the solutions in P by the non-ascending order of the objective function
    values f_g(p); Denote the sorted solutions as p_1, p_2, ..., p_|P|;
    Cong(p_1) = ∞, Cong(p_|P|) = ∞;
    for m = 2..|P − 1| do
        Cong(p_m) = Cong(p_m) + f_g(p_{m+1}) − f_g(p_{m−1});
    end for
end for
Return each Cong(p).
```

and $c_{k'}$ is the shortest among all the paths between $s_{i'}$ and the controllers (line 13). After switch $s_{i'}$ is mapped to a controller, the algorithm updates each list b_k by deleting the mapped switch $s_{i'}$ and the switches with more requests than the remaining capacity of controller c_k (line 14). When all the switches are mapped to the controllers, Algorithm 4 performs the operations of remap and swap to find other switch-controller association which can reduce the average switch-to-controller latency.

Operation remap(i, k, q) reassigns switch s_i from the originally mapped controller c_k to another controller c_q $(k \neq q)$ to generate a new association relationship, under the condition that controller c_q is not overloaded. The benefit of the remapping operation is defined by Eq. (9).

$$\pi_1(i, k, q) = l_{i,k} - l_{i,q} \tag{9}$$

where $\pi_1(i, k, q) > 0$ indicates that the switch-to-controller delay will be reduced, if switch s_i originally assigned to controller c_k is re-associated with controller c_q.

Operation swap(i, j) remaps switch s_i originally assigned to controller c_k to controller c_q, and reassigns switch s_j $(i \neq j)$ originally mapped to controller c_q to controller c_k $(k \neq q)$, under the condition that controllers c_k and c_q are not overloaded. The benefit of the swap operation is defined by Eq. (10). If $\pi_2(i, j, k, q) > 0$, we can decrease the switch-to-controller delay by swapping the mapping relationship between the switches and the controllers.

$$\pi_2(i, j, k, q) = (l_{i,k} + l_{j,q}) - (l_{i,q} + l_{j,k}) \tag{10}$$

4 Performance Evaluation

4.1 Simulation Setup

We evaluate the proposed algorithm MOCP against the state-of-the-arts: PSA [8] and EA [12]. The communication latency between two nodes in the network

Algorithm 4. *Switch Assignment*

Input:
$G = (V, E)$; K; Controller locations set $CL = \{c_1, c_2, \ldots, c_K\}$.
Output:
Mapping relationship between the controllers and the switches.
Each controller c_k maintains a list b_k which is initialized as $b_k = \phi$;
for each $c_k \in C$ **do**
Calculate the shortest path lengths between controller c_k to all the switches, and put all the switches in list b_k in the non-descending order of the path lengths;
end for
while $\exists\ b_k \neq \phi$ **do**
$k' = 0, i' = 0, l' = \infty$;
for $k = 1..K$ **do**
Select the head switch s_i of list b_k;
if $l_{i,k} < l'$ **then**
$l' = l_{i,k}, \quad i' = i, \quad k' = k$;
end if
end for
Assign switch $s_{i'}$ to controller $c_{k'}$;
Update each list b_k by deleting the mapped switch $s_{i'}$ and the switches with more requests than the remaining capacity of controller c_k;
end while
for $i = 1..|V|$ **do**
$k' = 0, \quad j' = 0, \quad h' = 0$;
for $j = i + 1..|V|$ **do**
Assume switches s_i and s_j are assigned to controllers c_k and c_q, respectively;
if $h' < \pi_2(i, j, k, q)$ **then**
$h' = \pi_2(i, j, k, q), \quad j' = j, \quad k' = q$;
end if
end for
if $h' > 0$ **then**
swap(i, j');
end if
end for
for $i = 1..|V|$ **do**
$k' = 0, \quad h' = 0$;
for $q = 1..K$ **do**
Assume s_i is mapped to c_k;
if $h' < \pi_1(i, k, q)$ **then**
$h' = \pi_1(i, k, q), \quad k' = q$;
end if
end for
if $h' > 0$ **then**
remap(i, k, k');
end if
end for

is approximated by the shortest path distance between the two nodes [13]. Two real network topologies of ATT and Internet2 [14] are used in the simulations. We also use network topology generator GT-ITM [15] to randomly generate two networks of Gnet1 and Gnet2. The processing capacity of the controllers is set as 1800 kilo-requests/s. The average number of requests from the switches is 200 kilo-requests/s, with the minimum being 150 kilo-requests/s and the maximum as 250 kilo-requests/s. The number of search times for the algorithms in the simulations is set as about 2.5% of the total feasible solution space [8]. The parameters of the networks and algorithm MOCP are shown in Table 2. We run each algorithm for 30 times, and merge the obtained non-dominant solution sets into a large set. The non-dominant solution set which has the largest intersection with the large set is taken the final result. We denote the solution sets obtained by algorithms MOCP, PSA and EA as P^M, P^P and P^E, respectively.

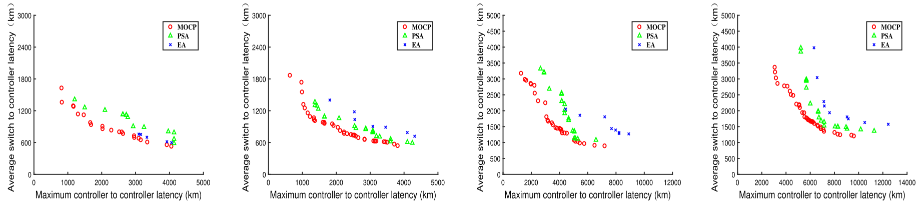
Table 2. Table of the parameters of networks and algorithm MOCP

Network	Number of nodes	Number of edges	Number of controllers	Number of iterations	Solution set size
ATT	25	57	4	6	50
Internet2	34	42	5	50	140
Gnet1	40	52	6	100	1000
Gnet1	51	64	7	3000	1000

The performance metrics are: Coverage (C) [16], Spacing (S) [17] and Maximum Spread (MS) [18], the number of solutions, and the optimal single-objective values. Coverage reflects the dominance relationship between two Pareto solution sets. Assuming P and Q are two Pareto optimal solution sets, the coverage of P over Q, $C(P,Q)$, is the ratio of the number of solutions which are in Q and dominated by the solutions in P to the total number of solutions in Q. $C(P,Q) = 1$ indicates that all solutions in Q are dominated by those in P. $C(P,Q) = 0$ indicates that no solution in Q is dominated by those in P. A large $C(P,Q)$ value shows that P is better than Q. Spacing evaluates the uniformity of the solution distribution in the Pareto optimal solution set. Assuming P and M are the Pareto solution set and the number of optimization objectives, respectively, $f_g(p)$ is the g-th objective function value of solution p. A smaller S value indicates that the solution set is more evenly distributed than a larger S value. Maximum spread measures the breadth of the solution distribution in the Pareto optimal solution set. The larger the MS value, the wider the distribution of the solution set.

4.2 Performance Evaluation of the Proposed Algorithm

Figures 1(a)–(d) depict the Pareto optimal solutions sets obtained by the three algorithms under the four networks. Algorithm MOCP achieves the best performance among the three algorithms. It is calculated that both $C(P^M,\ P^S)$



(a) ATT (b) Internet2 (c) Gnet1 (d) Gnet2

Fig. 1. Pareto sets under different networks.

and $C(P^M,\ P^E)$ are 1 in all the four networks, which demonstrates that the solutions of algorithm MOCP always dominate those of algorithms PSA and EA. Algorithm MOCP also leads to the smallest optimal single-objective values among the three algorithms. It can also be observed that algorithm MOCP gets more solutions than the other two algorithms, and the solutions of algorithm MOCP are more evenly distributed than those of algorithms PSA and EA.

Table 3. Table of performance of the algorithms

Network	Algorithm	Number of solutions	S	MS
ATT	MOCP	21	105	3422
	PSA	14	213	3046
	EA	5	113	978
Internet2	MOCP	37	87	3444
	PSA	22	123	2975
	EA	7	449	2594
Gnet1	MOCP	30	272	6341
	PSA	20	372	4522
	EA	8	574	4533
Gnet2	MOCP	37	165	6886
	PSA	20	378	6594
	EA	9	893	6593

Table 4. Table of optimal single-objective values of the algorithms

Network	Algorithm	Minimum $f_1(p)$ (km)	Minimum $f_2(p)$ (km)
ATT	MOCP	527	816
	PSA	585	1201
	EA	599	3094
Internet2	MOCP	539	634
	PSA	591	1370
	EA	718	1809
Gnet1	MOCP	892	1281
	PSA	1078	2651
	EA	1271	4436
Gnet2	MOCP	1202	3072
	PSA	1362	5221
	EA	1574	6310

Table 3 shows the simulation results of the three algorithms in different performance metrics. For the performance of the number of solutions, algorithm MOCP finds the largest size of optimal solution set among the three algorithms, and algorithm PSA obtains more solutions than algorithm EA. For the performance of spacing, algorithm MOCP outperforms algorithms PSA and EA in the four networks. The S value of algorithm MOCP is 26.8%–56.3% smaller than that of algorithm PSA. Algorithm MOCP algorithm achieves 7.1% better performance than algorithm EA in ATT network, while algorithm MOCP is about 80% better than algorithm EA in Internet2 and Gnet2 networks. It can be seen from the performance of the number of solutions and spacing metric that algorithm MOCP achieves better performance than algorithms PSA and EA in searching for local non-dominated solutions. For the performance of maximum spread, algorithm MOCP leads to bigger MS values than algorithms PSA and EA. A big MS value indicates that the solution set spreads across a large solution space. In Gnet1 network, the MS value of algorithm MOCP is 40.2% larger than that of algorithm PSA. In ATT network, algorithm MOCP achieves 249% larger MS value than algorithm EA. In Gnet2 network, the MS values of algorithms PSA and EA are similar, while algorithm MOCP obtains 4.4% larger MS value than algorithm PSA. From the performance of maximum spread and the number of solutions, it can be observed that algorithm MOCP outperforms algorithms PSA and EA in searching for global non-dominated solutions.

Table 4 lists the optimal single-objective values obtained by the three algorithms under different networks, which demonstrates that algorithm MOCP achieves smaller optimal single-objective values than algorithms PSA and EA in both of the objectives. Specifically, for the performance of the average switch-to-controller latency, algorithm MOCP is 8.7% and 17.2% better than algorithm PSA in the networks of Internet2 and Gnet1, respectively; while algorithm MOCP obtains 29.8% and 12% better results than algorithm EA in the networks of Gnet1 and ATT, respectively. For the performance of maximum inter-controller communication latency, algorithm MOCP performs 32% and 53% better than algorithm PSA in the networks of ATT and Internet2, respectively; while algorithm leads to 51.3% and 73.6% better results than algorithm EA in the networks of Gnet2 and ATT, respectively.

5 Conclusions

Both switch-to-controller latency and inter-controller communication delay have great impact on the network performance. In this paper, we formulated a novel multi-objective SDN controller placement problem with the objectives to minimize both the switch-to-controller and the inter-controller communication latency. We proposed an efficient metaheuristic-based Multi-Objective Controller Placement (MOCP) algorithm. We conducted experiments through simulations. Experimental results showed that algorithm MOCP could effectively reduce the latency between the controllers and from the switches to controllers simultaneously.

References

1. Cai, Z., Chen, Q.: Latency-and-coverage aware data aggregation scheduling for multihop battery-free wireless networks. IEEE Trans. Wirel. Commun. **20**(3), 1770–1784 (2021)

2. Chen, Q., Gao, H., Cai, Z., Cheng, L., Li, J.: Energy-collision aware data aggregation scheduling for energy harvesting sensor networks. In: IEEE Conference on Computer Communications (INFOCOM), pp. 117–125 (2018)
3. Chen, Q., Cai, Z., Cheng, L., Gao, H.: Low latency broadcast scheduling for battery-free wireless networks without predetermined structures. In: The 40th International Conference on Distributed Computing Systems (ICDCS), pp. 245–255 (2020)
4. Liao, J., Sun, H., Wang, J., Qi, Q., Li, K., Li, T.: Density cluster based approach for controller placement problem in large-scale software defined networkings. Comput. Netw. **112**, 24–35 (2017)
5. Yao, G., Bi, J., Li, Y., Guo, L.: On the capacitated controller placement problem in software defined networks. IEEE Commun. Lett. **18**(8), 1339–1342 (2014)
6. Qin, Q., Poularakis, K., Iosifidis, G., Tassiulas, L.: SDN controller placement at the edge: optimizing delay and overheads. In: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, pp. 684–692. IEEE (2018)
7. Hock, D., Hartmann, M., Gebert, S., Jarschel, M., Zinner, T., Tran-Gia, P.: Pareto-optimal resilient controller placement in SDN-based core networks. In: Proceedings of the 2013 25th International Teletraffic Congress (ITC), pp. 1–9. IEEE (2013)
8. Lange, S., et al.: Heuristic approaches to the controller placement problem in large scale SDN networks. IEEE Trans. Netw. Serv. Manag. **12**(1), 4–17 (2015)
9. Fan, Y., Ouyang, T.: Reliability-aware controller placements in software defined networks. In: The 21st IEEE International Conference on High Performance Computing and Communications (HPCC), pp. 2133–2140. IEEE (2019)
10. Fan, Y., Wang, L., Yuan, X.: Controller placements for latency minimization of both primary and backup paths in SDNs. Comput. Commun. **163**, 35–50 (2020)
11. Levin, D., Wundsam, A., Heller, B., Handigol, N., Feldmann, A.: Logically centralized? State distribution trade-offs in software defined networks. In Proceedings of the First Workshop on Hot Topics in Software Defined Networks, pp. 1–6 (2012)
12. Zhang, T., Bianco, A., Giaccone, P.: The role of inter-controller traffic in SDN controllers placement. In: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 87–92. IEEE (2016)
13. Heller, B., Sherwood, R., McKeown, N.: The controller placement problem. ACM SIGCOMM Comput. Commun. Rev. **42**(4), 473–478 (2012)
14. Knight, S., Nguyen, H.X., Falkner, N., Bowden, R., Roughan, M.: The Internet topology zoo. IEEE J. Sel. Areas Commun. **29**(9), 1765–1775 (2011)
15. Thomas, M., Zegura, E.W.: Generation and analysis of random graphs to model internetworks. Tech. rep., Georgia Institute of Technology (1994)
16. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)
17. Schott, J.R.: Fault tolerant design using single and multicriteria genetic algorithm optimization. Tech. rep., Air force inst of tech Wright-Patterson afb OH (1995)
18. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. **8**(2), 173–195 (2000)