






A Scalable and Reliable Model for the Placement of Controllers in SDN Networks

Mohammad Ashrafi^(✉) , Noélia Correia^{}, and Faroq AL-Tam^{}

Center for Electronic, Optoelectronic and Telecommunications (CEOT),
Faculty of Science and Technology, University of Algarve, 8005-139 Faro, Portugal
mohammad.ashrafi@gmail.com, {ncorreia,ftam}@ualg.pt

Abstract. In this article, a mathematical model is developed to place controllers in multi-controller software-defined networking (SDN), while considering: resilience, scalability, and inter-plane latency. The model proved to be effective since it is able to provide resilient solutions under different fail-over scenarios, while at the same time avoid working close to the capacity limits of controllers, which offers a scalable model for multi-controller SDN.

Keywords: Control placement · Software Defined Networking
Reliability · Scalability

1 Introduction

Software Defined Networking (SDN) is an emergent paradigm that offers a software-oriented network design, simplifying network management by decoupling the control logic from forwarding devices [10]. The SDN is composed of three planes: management, control, and data. The management plane is responsible for defining the network policies, and it is connected to the control plane via northbound interfaces. The brain of SDN is the control plane, which interacts with the data plane using southbound interfaces like OpenFlow [5].

Most of the current available controllers, like NOX [6] and Beacon [4], are physically centralized. Although a single controller offers a complete network-wide view, it represents a single point of failure and lacks both reliability and scalability [19]. For this reason multi-controller SDNs were developed [5], allowing the control plane to be physically distributed, but maintaining it logically centralized by synchronizing the network state among controllers [20]. Controllers of this type include OpenDaylight [11] and Kandoo [7]. Multi-controller SDNs

This work was developed within the Centre for Electronic, Optoelectronic and Telecommunications (CEOT), and supported by the D/MULTI/00631/2013 project of the Portuguese Science and Technology Foundation (FCT).

are able to solve the main problems found in the centralized SDN, but new challenges are introduced, like network state synchronization and switch-controller mappings. Another main problem in multi-controller SDNs is the Controller Placement Problem (CPP) [15]. The problem is proved to be NP-hard [17], and is one of the hottest topics in multi-controller SDNs [8, 12, 13, 17].

When a switch receives a new packet, it consults its forwarding rules in the flow table in order to determine how to handle the packet. If there is no match in the flow table, the packet is buffered temporarily and the switch initiates a **packet-in** message to the responsible controller. Reactively, the controller calculates the path for this packet and installs a new rule in the affected switches. Two major factors that influence the effectiveness of this process are: (i) load at the controller; (ii) propagation delay between the switch and the controller. These two constitute a single efficiency measure: the flow setup time, which will be twice the propagation delay between switch and controller plus the queuing and processing time of the packet in the controllers.

The CPP aims at deciding the number of required controllers and where to place them [8], partitioning the network into subregions (domains), while considering some quality criteria and cost constraints [1, 12]. The CPP model discussed in this article incorporates the previously mentioned flow setup time, while presenting reliable and scalable solutions.

The remaining part of this paper is organized as follows. Section 2 discusses work related with CPP in SDNs. Section 3 discusses the mathematical model, whose results are discussed in Sect. 4. Finally, some conclusions are drawn in Sect. 5, together with future work.

2 Related Work

The placement problem was mentioned for the first time in [8]. In fact, this problem is similar to the popular facility location problem, and is solved in the aftermentioned article as K-center problem, to minimize the inter-plane propagation delay. In [17] the problem is extended to incorporate the capacity of the controllers. A new metric called expected percentage of control path loss is proposed in [16] to guarantee a reliable model. Cost, controller type, bandwidth, and other factors are considered in [12], and the expansion problem is considered in [13]. The problem is usually modeled in these articles as integer programming.

Heuristic methods that incorporate switch migration can be found in [18]. In [2] a game model is also proposed. A comprehensive review of heuristic methods can be found in [9]. QoS-aware CPP is presented in [3] and solved using greedy and network partitioning algorithms. Recently, scalability and reliability issues in large-scale networks are considered in [1]. Clustering and genetic approaches are proposed, but these approaches are prone to sub-optimality.

When comparing our model with other in the literature, our model determines the optimal placement considering different failure scenarios and latency to reduce latency and overload of controllers while ensuring scalability, leading to reduced overload at controllers.

Table 1. Known information.

Term	Description
\mathcal{C}	Set of controllers
\mathcal{N}_c	Possible places for $c \in \mathcal{C}$, $\mathcal{N}_c \subset \mathcal{N}$
h_c	Number of requests per second that can be handled by controller $c \in \mathcal{C}$
\mathcal{S}	Set of switches
p_s	Number of requests not matching the lookup table of $s \in \mathcal{S}$
\mathcal{F}	Set of physical link failure scenarios. Includes a scenario where all links are up
\mathcal{L}_f	Set of physical links failing when scenario $f \in \mathcal{F}$ occurs

Table 2. Required variables.

Variable	Description
σ_n^c	One if controller $c \in \mathcal{C}$ is placed at location $n \in \mathcal{N}_c$
$\mu_{n_i, n_j, f, l}^{c_i, c_j}$	One if link $l \in \mathcal{L} \setminus \mathcal{L}_f$ is used for inter-controller $c_i - c_j$ communication, located at nodes n_i and n_j respectively, when failure $f \in \mathcal{F}$ occurs
β_f^l	One if link $l \in \mathcal{L} \setminus \mathcal{L}_f$ is used for inter-controller communication when failure $f \in \mathcal{F}$ occurs
$\gamma_f^{s, c}$	One if switch $s \in \mathcal{S}$ is assigned to controller $c \in \mathcal{C}$ when failure $f \in \mathcal{F}$ occur
$\lambda_{n, f}^{s, c}$	One if switch $s \in \mathcal{S}$ is assigned to controller $c \in \mathcal{C}$ when failure $f \in \mathcal{F}$ occurs, and the controller is placed at location $n \in \mathcal{N}_c$
$\phi_{f, l}^{s, c, n_i}$	One if switch $s \in \mathcal{S}$ is assigned to controller $c \in \mathcal{C}$ located at nodes $n_i \in \mathcal{N}_c$ when failure $f \in \mathcal{F}$, and uses link $l \in \mathcal{L} \setminus \mathcal{L}_f$ in its path
δ	Scalability factor, $0 \leq \delta \leq 1$
Θ^{TOTAL}	Total latency, under any failure scenario
Π^{TOTAL}	Total number of links used, under any failure scenario

3 Mathematical Model

In the following discussion the physical topology graph is assumed to be defined by $\mathcal{G}(\mathcal{N}, \mathcal{L})$, where \mathcal{N} is a set of physical nodes/locations and \mathcal{L} is a set of physical links. The remaining notation for known information and variables, used through this paper, is presented in Tables 1 and 2.

Objective Function: To ensure the linear scale up of the SDN network the goal will be:

$$\text{Minimize } \delta + K_1 \frac{\Theta^{\text{TOTAL}}}{\Delta} + K_2 \frac{\Pi^{\text{TOTAL}}}{\Delta} \quad (1)$$

where Δ is a big value. The primary goal is to minimize δ , and then to reduce latency. The factors K_1 and K_2 should be adapted according to inter-controller

and switch-controller latency relevance. The motivation behind giving δ more importance is that the provided solution for controller placement will be used for a relatively long period of time, during which traffic conditions may change. Therefore, the scalability of the solution is considered to be critical.

Constraints: The following additional constraints must be fulfilled.

– Placement of controllers:

$$\sum_{\{n \in \mathcal{N}_c\}} \sigma_n^c = 1, \forall c \in \mathcal{C} \quad (2)$$

$$\begin{aligned} & \sum_{\{l \in \mathcal{L} \setminus \mathcal{L}_f : \text{src}(l) = n\}} \mu_{n_i, n_j, f, l}^{c_i, c_j} - \sum_{\{l \in \mathcal{L} \setminus \mathcal{L}_f : \text{dst}(l) = n\}} \mu_{n_i, n_j, f, l}^{c_i, c_j} = \\ & = \begin{cases} \sigma_{n_i}^{c_i}, & \text{if } n = n_i \\ -\sigma_{n_j}^{c_j}, & \text{if } n = n_j, \forall c_i, c_j \in \mathcal{C}, \forall f \in \mathcal{F}, \forall n_i \in \mathcal{N}_{c_i}, \forall n_j \in \mathcal{N}_{c_j}, \forall n \in \mathcal{N} \\ 0, & \text{otherwise} \end{cases} \quad (3) \end{aligned}$$

Constraints (2) ensure a single location for each controller $c \in \mathcal{C}$, while constraints (3) ensure that there will be a path between every pair of controllers, under any failure scenario, while considering their location. These paths are used for state synchronization.

In [14] it is stated that the controller load can be reduced, achieving load balance among neighboring controllers, if controller needs to communicate only with its local neighbors. Therefore, the paths from any controller, towards all the other controllers, should share as many links as possible (leads to a bus logical topology). This is ensured by the following constraints.

$$\beta_f^l \geq \mu_{n_i, n_j, f, l}^{c_i, c_j}, \forall c_i, c_j \in \mathcal{C}, \forall n_i, n_j \in \mathcal{N}_c, \forall f \in \mathcal{F}, \forall l \in \mathcal{L} \quad (4)$$

$$\Pi^{\text{TOTAL}} = \sum_{\forall f \in \mathcal{F}} \sum_{\forall l \in \mathcal{L}} \beta_f^l \times 1/2 \quad (5)$$

where Π^{TOTAL} , counting for the highest number of end-to-end hops in inter-controller communication, is to be included in the objective function.

– Switch to controller mapping:

$$\sum_{\{c \in \mathcal{C}\}} \gamma_f^{s, c} = 1, \forall s \in \mathcal{S}, \forall f \in \mathcal{F} \quad (6)$$

$$\sum_{\{s \in \mathcal{S}\}} \gamma_f^{s, c} \times p_s \leq h_c \times \delta, \forall c \in \mathcal{C}, \forall f \in \mathcal{F} \quad (7)$$

Constraints (6) ensure single mapping, and constraints (7) avoid the overload of controllers, while ensuring scalability regarding future switch migrations (triggered to deal with load fluctuations) due to the use of δ , which is, be included in the objective function too. Again, the multiple failure scenarios are taken into consideration.

– Switch-controller latency:

$$\lambda_{n,f}^{s,c} \geq \gamma_f^{s,c} + \sigma_n^c - 1, \forall c \in \mathcal{C}, \forall n \in \mathcal{N}_c, \forall f \in \mathcal{F}, \forall s \in \mathcal{S} \quad (8)$$

$$\begin{aligned} & \sum_{\{l \in \mathcal{L} \setminus \mathcal{L}_f : \text{src}(l) = n\}} \phi_{f,l}^{s,c,n_i} - \sum_{\{l \in \mathcal{L} \setminus \mathcal{L}_f : \text{dst}(l) = n\}} \phi_{f,l}^{s,c,n_i} = \\ & = \begin{cases} \lambda_{n,f}^{s,c}, & \text{if } n = \text{loc}(s) \\ -\lambda_{n,f}^{s,c}, & \text{if } n = n_i, \\ 0, & \text{otherwise} \end{cases} \quad \forall s \in \mathcal{S}, \forall c \in \mathcal{C}, \forall f \in \mathcal{F}, \forall n_i \in \mathcal{N}_c, \forall n \in \mathcal{N} \end{aligned} \quad (9)$$

where $\text{loc}(s)$ is the location of switch s . The total latency is obtained by:

$$\Theta^{\text{TOTAL}} = \sum_{\{s \in \mathcal{S}\}} \sum_{\{c \in \mathcal{C}\}} \sum_{\{f \in \mathcal{F}\}} \sum_{\{l \in \mathcal{L}\}} \sum_{\{n \in \mathcal{N}_c\}} \phi_{f,l}^{s,c,n} \quad (10)$$

Θ^{TOTAL} is included in the objective function for latency minimization.

– Non-negativity assignment to variables:

$$0 \leq \delta \leq 1; \sigma_n^c, \mu_{n_i, n_j, f, l}^{c_i, c_j}, \beta_f^l, \gamma_f^{s,c}, \lambda_{n,f}^{s,c}, \phi_{f,l}^{s,c,n_i} \in \{0, 1\}; \Theta^{\text{TOTAL}}, \Pi^{\text{TOTAL}} \in \mathbb{R}^+. \quad (11)$$

This model assumes that the physical layer is not disconnectable under a single physical link failure. The CPLEX¹ optimizer has been used to solve the problem instances discussed in the following section.

4 Results

The values for input parameters, used by the optimizer, are displayed in Table 3. Different failure cases were used to evaluate the model under three different topologies (Fig. 1). A case relates to single link failure (no two links fail at the same time in each scenario), while the other relates to multiple link failure scenarios. Two percentages for affected links were tested. More specifically:

Case I: Single link failure scenarios, where $\cup_{f \in \mathcal{F}} \mathcal{L}_f$ affects a total of 5% (a) or 15% (b) of all the links;

Case II: Two or more links failing simultaneously, in each failure scenario, where $\cup_{f \in \mathcal{F}} \mathcal{L}_f$ affects a total of 5% (a) or 15% (b) of all the links.

¹ IBM ILOG CPLEX Optimizer version 12.8.

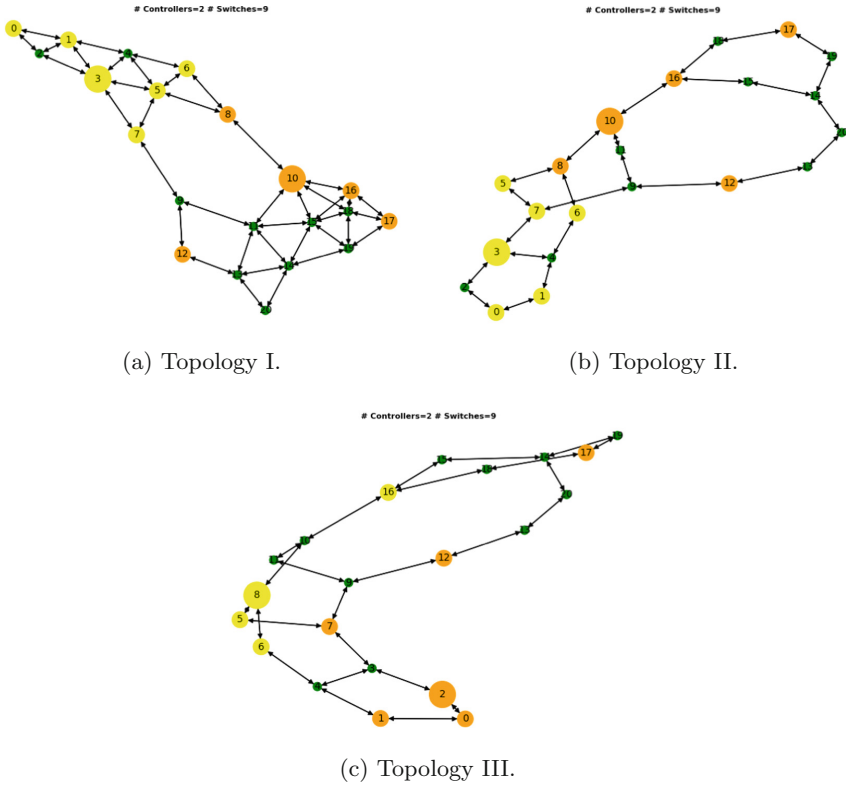


Fig. 1. Physical topologies used for analysis of results. Large nodes are controllers, medium are switches, and small are non-used locations.

Table 3. Input parameters.

Parameter	Value
K_1	0.5
K_2	0.5
p_s	[40, 100]
h_s	[500, 600]

That is, in case Ia 5% of the links may fail, but no two links fail at the same time, while in case IIa there is a 0.5 probability for any pair of links to go down at the same time, which may lead to failure scenarios where two or more links fail simultaneously. In cases Ib and IIb, 15% of the links may fail.

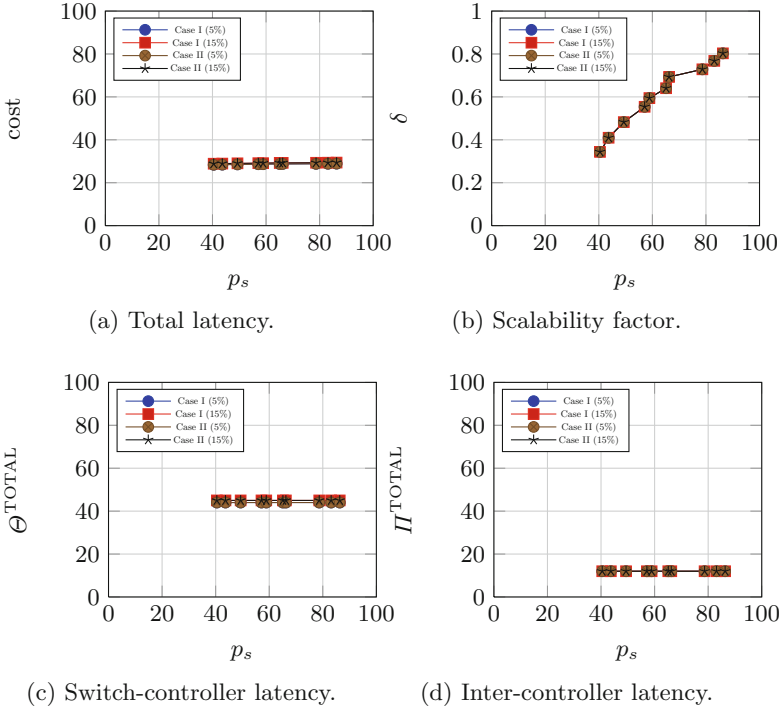


Fig. 2. Results for Topology I: high connectivity and relatively low number of possible locations for the controllers.

Table 4. Physical topology details.

Topology	# Nodes	# Links	# Possible controller locations
Topology I	21	40	4
Topology II	21	25	4
Topology III	21	25	9

Topology I: Results for this topology are shown in Fig. 2. This is the most dense topology, with relatively small number of possible locations for controllers. The scalability factor δ increases linearly as the number of `packet-in` messages increases.

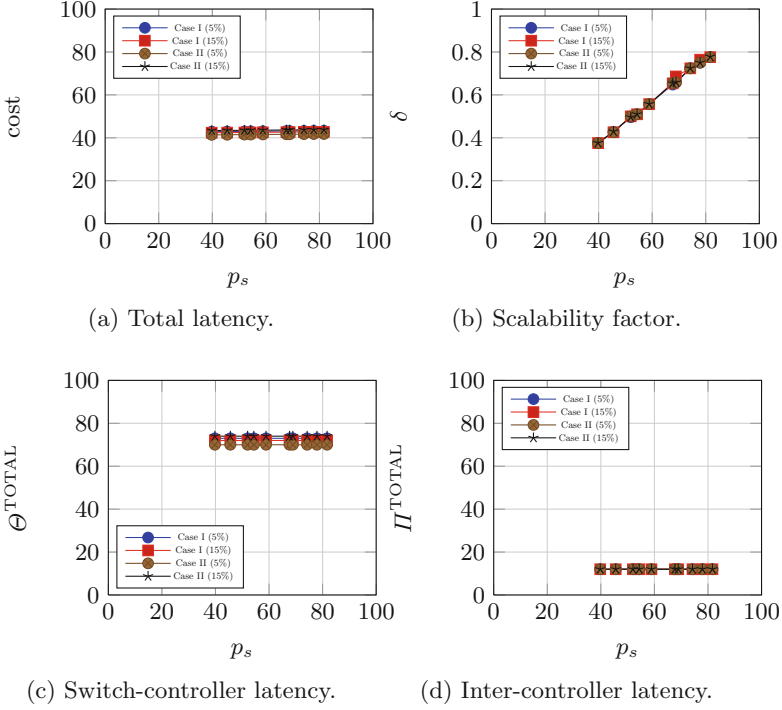


Fig. 3. Results for Topology II: low connectivity and relatively low number of possible locations for the controllers.

Topology II: Results for this topology are depicted in Fig. 3. This topology presents less links than Topology I, but the number of possible locations are kept similar. The main difference regarding these results is that the switch-controller latency has increased.

Topology III: Results for this topology are shown in Fig. 4. This topology also presents less links than Topology I, but the number of possible locations is increased. In this case it is possible to observe that the total latency significantly decreases. Therefore, the model was able compensate the reduced number of links, finding adequate places for controllers that lead to lower latency.

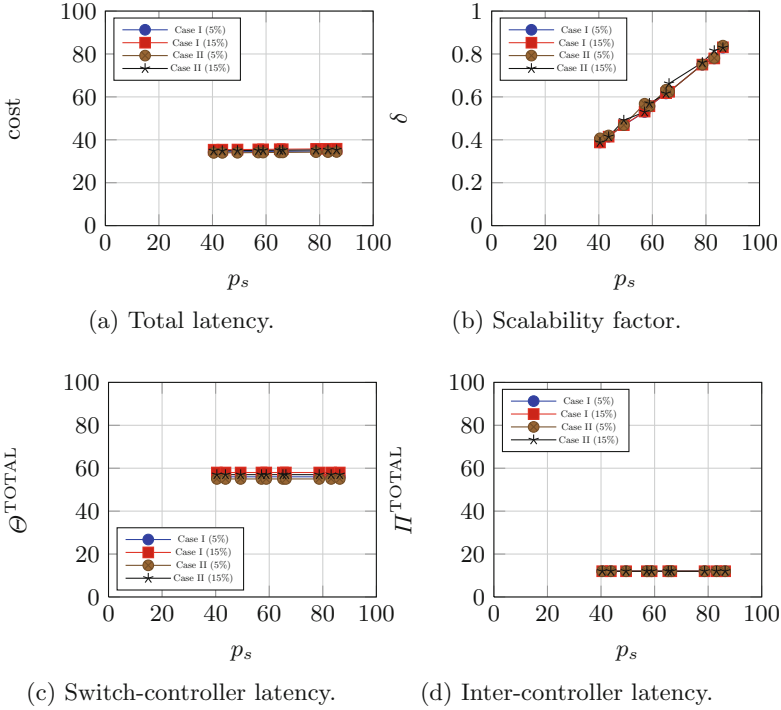


Fig. 4. Results for Topology III: low connectivity and relatively high number of possible locations for the controllers.

5 Conclusions

In this paper, a scalable and reliable model for controller placement is introduced. This model is mathematically formulated and optimal solutions for controller placement, under different failure scenarios, are obtained. Results show that scalability is ensured under different failure scenarios, while latency increase can be compensated through more freedom in controller’s locations. The model also serves adequately multiple failure scenarios, presenting similar results for more critical failure scenarios and less critical ones.

In general, results show that the model is able to keep scalability (δ) while considering failure scenarios, ensuring load balancing among controllers. The latency may increase when less network connectivity decreases, but this might be avoided if more possible locations for controllers are allowed. Results are similar for Cases Ia, Ib, IIa and IIb, meaning that the model makes a controller placement that serves adequately multiple failure scenarios.

References

1. Bannour, F., Souihi, S., Mellouk, A.: Scalability and reliability aware SDN controller placement strategies. In: 13th International Conference on Network and Service Management (CNSM), pp. 1–4, November 2017
2. Cheng, G., Chen, H.: Game model for switch migrations in software-defined network. *Electron. Lett.* **50**(23), 1699–1700 (2014)
3. Cheng, T.Y., Wang, M., Jia, X.: QoS-guaranteed controller placement in SDN. In: IEEE Global Communications Conference (GLOBECOM) (2015)
4. Erickson, D.: The beacon OpenFlow controller. In: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, HotSDN 2013, pp. 13–18. ACM, New York (2013)
5. Open Networking Foundation: OpenFlow switch specification. Technical report ONF TS-025 (2015)
6. Gude, N., et al.: NOX: towards an operating system for networks. *SIGCOMM Comput. Commun. Rev.* **38**(3), 105–110 (2008)
7. Hassas Yeganeh, S., Ganjali, Y.: Kandoo: a framework for efficient and scalable offloading of control applications. In: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, HotSDN 2012, pp. 19–24. ACM, New York (2012)
8. Heller, B., Sherwood, R., McKeown, N.: The controller placement problem. *ACM SIGCOMM Comput. Commun. Rev.* **42**(4), 473–478 (2012). <https://doi.org/10.1145/2377677.2377767>
9. 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)
10. McKeown, N., et al.: OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Comput. Commun. Rev.* **38**(2), 69–74 (2008)
11. Medved, J., Tkacik, A., Varga, R., Gray, K.: OpenDaylight: towards a model-driven SDN controller architecture. In: IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM). IEEE (2014)
12. Sallahi, A., St-Hilaire, M.: Optimal model for the controller placement problem in software defined networks. *IEEE Commun. Lett.* **19**(1), 30–33 (2015)
13. Sallahi, A., St-Hilaire, M.: Expansion model for the controller placement problem in software defined networks. *IEEE Commun. Lett.* **21**(2), 274–277 (2017)
14. Schmid, S., Suomela, J.: Exploiting locality in distributed SDN control. In: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, HotSDN 2013, pp. 121–126. ACM, New York (2013)
15. Vizarrreta, P., Machuca, C.M., Kellerer, W.: Controller placement strategies for a resilient SDN control plane. In: Jonsson, M., Rak, J., Somani, A., Papadimitriou, D., Vinel, A. (eds.) Proceedings of 2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM), pp. 253–259 (2016)
16. Yannan, H., Wendong, W., Xiangyang, G., Xirong, Q., Shiduan, C.: On reliability-optimized controller placement for software-defined networks. *China Commun.* **11**(2), 38–54 (2014)
17. 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)

18. Yao, L., Hong, P., Zhang, W., Li, J., Ni, D.: Controller placement and flow based dynamic management problem towards SDN. In: IEEE International Conference on Communication Workshop (ICCW), pp. 363–368. IEEE (2015)
19. Yeganeh, S.H., Tootoonchian, A., Ganjali, Y.: On scalability of software-defined networking. *IEEE Commun. Mag.* **51**(2), 136–141 (2013)
20. Zhang, Y., Cui, L., Wang, W., Zhang, Y.: A survey on software defined networking with multiple controllers. *J. Netw. Comput. Appl.* **103**, 101–118 (2018)