AUT Journal of Mathematics and Computing

AUT J. Math. Comput., 3(2) (2022) 185-192
DOI: 10.22060/AJMC.2022.21706.1100

Original Article

Survivable controller placement in software defined network

Ahmad Moradi*[a], Ali AbdiSeyedkolaei[a]

[a]*Department of Mathematical Sciences, University of Mazandaran, Babolsar, Iran*

ABSTRACT: One of the problems raised in software defined networks is to determine the number and installation location of controllers so that the cost of implementation reduced and survivability of the network against link or node failure increased. Current investigation in SDN imposes full mesh topology in order to connect controllers. This approach while incurring a huge installation cost, dose not carefully incorporate network survivability requirements. In this paper, we improve an existing integer programming approach to a novel model so as to effectively address user defined survivability requirements. Computational results reported also reveals that our models could be solved by state-of-the-art MIP solvers like CPLEX within a reasonable time limit.

Review History:
Received:20 June 2022
Accepted:23 August 2022
Available Online:01 September 2022

Keywords:
Controller placement
Mixed integer programming
Survivablility
Software defined network

AMS Subject Classification (2010):
90B10; 68M10

(Dedicated to Professor S. Mehdi Tashakkori Hashemi)

1. Introduction

An SDN network provides the ability to reach a programmable network by separating control panel from data panel so as to improve network performance [3, 6]. This separation provides benefits such as simple network management, improved network performance, and network innovation. The control panel provides necessary information for routing in the network. The data panel has the task of transferring packets based on the information contained in its routing table. In SDN networks, the isolated control panel placed on the server or application is called controller. The data panel remains in a switch or a router as the port forwarding. The control panel is responsible for managing the data panel and it is often responsible for flow propagation in the network, by assigning input flows to the switches [7]. This gives the controller a pivotal role because it allows you to have a complete knowledge of the network in optimizing flow management and supporting user requirements [8]. It will motivate a variant of location problem called controller placement problem. In this problem, the goal is to determine the location of the controllers and their required number in the network, so that the total cost of installing controllers, the cost of connecting switches to controllers and the cost of connecting controllers to each other is minimized [1]. In terms of computational complexity, the problem belongs to the class of NP-hard problems [12]. A reader interested in more details on SDN concepts is referred to [5, 9].

In this paper, we make use of mathematical programming approach and first redefine the controller placement problem in software defined networks as a mixed integer nonlinear programming formulation with the capability to

*Corresponding author.
E-mail addresses: a.moradi@umz.ac.ir, a.abdi@umz.ac.ir

impose a general connected topology among controllers. Then we reduce the formulation to a mixed integer linear program as to be able to solve more efficiently where we also incorporate user defined survivability requirement to the mixed integer programming formulation. Experiments also admit that the provided formulation designs networks of much less installation cost while accepting a general connected topology among controllers as well as user defined survivability parameters.

Related research efforts has been performed in [1, 7, 8, 11]. More specifically, authors in [11]considered controller load balancing minimize latency while not considering controller failure. Authors in [7] provided integer linear programming formulation to reduce implementation cost as well as controlling the notion of latency but not survivability. Research paper [8] takes a clustering approach to controller placement where the failure of a cluster head controller is still an issue not considered. Paper [1] also provide load balancing and latency minimization but it lacks of cost optimization and survivability control. We also admit that part of the results presented here are reported in [2, 10].

The remainder of this article is organized as follows: Section 2 and 3 describe mathematical formulations of the problem. Section 4 analyzes the results of the simulation of the proposed formulation on all instances in comparison with the full mesh model represented in [7]. Finally, conclusions obtained from the calculations are expressed in section 5.

2. Problem Definition and Formulations

In this section we carefully define the Controller Placement problem as a mixed integer optimization problem. Let's denote the underlying network with $G = (N, A)$ where N and A are the set of network nodes and network arcs, respectively. We assume without loss of generality that a node could contain only a switch or a controller. Let us denote the set of switches with S and the set of controllers with F where $S \cap F = \emptyset$. In our model, we need the following notations to capture controller to controller links and switch to controller links separately:

$$A_F = \{ij \in A \mid i, j \in F\},$$
$$A_S = \{ij \in A \mid i \in S, j \in F\},$$
$$P = \{(i,j) : i \in F, j \in F, i < j\}.$$

For each switch, s, the number of packets that do not match on the switch's lookup table and that are sent to the future connected controller is shown with parameter β. For each controller, c, the parameters μ^c and α^c, indicate port limit and processing capacity of the controller respectively. Other parameters used in our formulations are the number of available controllers δ^c and γ^c the installation cost of a controller. Define the following decision variables:

$$x_{ij} = \begin{cases} 1 & \text{If arc } (i,j) \text{ is selected,} \\ 0 & \text{Otherwise} \end{cases}$$

$$z_i^c = \begin{cases} 1 & \text{If a controller of type } c \text{ is placed in node } i, \\ 0 & \text{Otherwise} \end{cases}$$

$$g_{ij}^{pq} = \begin{cases} 1 & \text{If a unit flow from location } p \text{ to location } q \text{ passes arc } (i,j), \\ 0 & \text{Otherwise} \end{cases}$$

A mathematical formulation of the controller placement problem is given as the following:

$$\text{Formulation (1)} : \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{c \in C} \gamma^c \sum_{i \in F} z_i^c$$

$$\sum_{ij \in A_F} g_{ij}^{pg} - \sum_{ji \in A_F} g_{ji}^{pg} = \begin{cases} \sum_{c \in C} (z_p^c * z_q^c) & i = p \\ -\sum_{c \in C} (z_p^c * z_q^c) & i = q \\ 0 & i \neq p, q \end{cases}, \forall i \in F, \forall pq \in P \tag{1}$$

$$\sum_{j \in F, ij \in A_S} x_{ij} = 1, \forall i \in S \tag{2}$$

$$x_{ij} \leq \sum_{c \in C} z_j^c, \forall ij \in A_S, i \in S, j \in F \tag{3}$$

$$\sum_{c \in C} z_i^c \leq 1, \forall i \in F \tag{4}$$

$$\sum_{i\in F} z_i^c \leq \delta^c, \quad \forall c \in C \tag{5}$$

$$\sum_{i<j, j\in F} x_{ij} + \sum_{j\in S, ji\in A_S} x_{ji} \leq \sum_{c\in C} \mu^c * z_i^c, \forall i \in F \tag{6}$$

$$\sum_{j\in S, ji\in A_S} \beta * x_{ji} \leq \sum_{c\in C} \alpha^c * z_i^c, \forall i \in F \tag{7}$$

$$x_{ij} \in \{0,1\}, \forall ij \in A \tag{8}$$

$$z_i^c \in \{0,1\}, \forall i \in F, c \in C \tag{9}$$

$$t_{pq} \in \{0,1\}, \forall pq \in P \tag{10}$$

$$g_{ij}^{pq} \in \{0,1\}, \forall ij \in A_F, pq \in P \tag{11}$$

The objective function consists of two parts. The first part measures the installation cost of a controller and the second part measures costs incurred by connecting controllers to controllers and switches to controllers. Constraint (1) assures that every two locations, each containg a controller, are connected to each other. Constraints (2) and (3) state that each switch could only be assigned to a location that contains a controller. Constraint (4) states that at most one controller could be installed in a location. Constraints (5)-(7) restrict the numbers of available controllers, the number of ports and processing capacity of a controller respectively. Other constraints simply show binary nature of decision variables. Further explanation on the constraints could be found in [2, 10].

The term $\sum_{c\in C}(z_p^c * z_q^c)$, in equation (1) makes this constraint a nonlinear one. In order to make it linear we use *McCormick constraints* as the following. Define, $t_{pq} = \sum_{c\in C}(z_p^c * z_q^c)$ Each t_{pq} is a binary variable indicating whether or not locations p and q contain a controller simultaneously. The above equation could be replaced with:

$$\begin{cases} t_{pq} \leq \min(\sum_{c\in C} z_p^c, \sum_{c\in C} z_q^c) \\ t_{pq} \geq \max(0, \sum_{c\in C} z_p^c - (1 - \sum_{c\in C} z_q^c)) \end{cases}, \forall pq \in P$$

Or equivalently:

$$\begin{cases} t_{pq} \leq \sum_{c\in C} z_p^c \\ t_{pq} \leq \sum_{c\in C} z_q^c \\ t_{pq} \geq \sum_{c\in C} z_p^c - (1 - \sum_{c\in C} z_q^c) \end{cases}, \forall pq \in P$$

As a result, we have:

$$\text{Formulation (2): } \min \sum_{ij\in A} x_{ij} c_{ij} + \sum_{c\in C} \gamma^c \sum_{i\in F} z_i^c$$

$$\sum_{ij\in A_F} g_{ij}^{pg} - \sum_{ji\in A_F} g_{ji}^{pg} = \begin{cases} t_{pq} & i = p \\ -t_{pq} & i = q \\ 0 & i \neq p, q \end{cases}, \forall i \in F, \forall pq \in P \tag{12}$$

$$\begin{cases} t_{pq} \leq \sum_{c\in C} z_p^c \\ t_{pq} \leq \sum_{c\in C} z_q^c \\ t_{pq} \geq \sum_{c\in C} z_p^c - (1 - \sum_{c\in C} z_q^c) \end{cases}, \forall pq \in P \tag{13}$$

$$(2) - (11)$$

3. Survivable Controller Placement

Since network failures that disconnect the control and data planes could lead to severe packet loss and performance degradation [4], it is of great importance to improve survivability in context of controller placement problem consider the following cases:

Case 1. Switch to controller link failure: Define $\zeta_i = 1$ as to be a user defined parameter indicating the number of proper backup controllers connected to switch i. One could replace the following set of constraints with equation (2) in formulation (1) in order to assure that the network remains connected even after ζ_i switches to controller link failure.

$$\sum_{j \in F, ij \in A_S} x_{ij} = \zeta_i, \quad \forall i \in S \qquad (2)'$$

Case 2. Controller to controller link failure: In this case, we have to use the notion of *arc disjoint paths*. Consider locations p and q each containing a controller. Two paths from p to q over G are said to be arc disjoint if and only if they don't have any edge in common. Now consider a user defined parameter η_{pq} as to be the number of proper arc disjoint paths from p to q. In order to impose existence of at least η_{pq} paths between location p and q update (12) as to be:

$$\sum_{ij \in A_F} g_{ij}^{pg} - \sum_{ji \in A_F} g_{ji}^{pg} = \begin{cases} \eta_{pq} * tpq & i = p \\ -\eta_{pq} * tpq & i = q \\ 0 & i \neq p, q \end{cases} , \forall i \in F, \forall pq \in P \qquad (12)'$$

imposing

$$g_{ij}^{pg} + g_{ji}^{pg} \leq x_{ij}, \forall pq \in P, \forall ij, ji \in A_F \qquad (14)$$

assures edge-disjointness of such paths.

Case 3. Controller failure In this case, switches connected to the failed controller could be reassigned by means of equation $(2)'$. In order to backup paths between controllers we have to use the notion of node disjoint paths. Two paths between locations p and q are called node disjoint if they do not have any node in common except the nodes p and q. Node disjoint paths in G are equivalent to arc disjoint paths in a transformed network,$G' = (S' \cup F', E')$, constructed as the following. Having $G = (S \cup F, E)$

- add each s in S to S'
- for each $p \in F$, add two copies p_{in} and p_{out} to F'
- for each link ij in A_S, add ij_{in} to E'
- for each link ij in A_F, add $i_{out}j_{in}$ to E'
- for each p in F, add $p_{in}p_{out}$ to E'

As a result, constructing a survivable network against node-failure is possible by solving the above formulations over the transformed network G'. Let the formulation (2) while all survivability constraints are replaced, be called formulation (3).

4. The Simulation Results

In this section, we evaluate our mixed integer programming formulations against the existing full-mesh formulations reported in [7]. Here, quality as well as survivability of the solutions found by each of the formulations are compared. For each instance, the network topology is randomly extracted from a 1000×1000 grid. It means each node of the grid with probability p_r is a node of network graph. The value of p_r is set to 0.5. After selecting the nodes, the complete graph induced by the selected nodes is considered. On this graph, weight of an edge is defined by the Euclidean distance between its end nodes. In the next step, the nodes containing a switch on the graph are specified. For an instance with i switches, i nodes of the graph are randomly selected, and on each of them a switch is installed. The parameter i, is taken from $\{10, 20, 30, 40, 50, 75, 100, 150, 200\}$. The parameter j, the number of possible locations for controller installation, is taken from $\{10, 15, 20\}$.

Table 1: Problem solving parameters as in [6]

Parameter name	Value		
	Type 1 Controller	Type 2 Controller	Type 3 Controller
Cost per controller (γ^c)	1200$	2500$	6500$
Number of ports per controller(μ^c)	8	16	32
Processing Capacity by Controller(α^c)	2500	4000	8000
Number of available controllers(δ^c)	20	15	10
Link cost per meter		8.25$	
Packet size(β)		150 Byte	

All computations of this section are done on an Intel core i5 under Windows operating system with 8 GB of main memory. For CPLEX, the time limit is set to 7200 seconds. CPLEX solution parameters are given, as in [6], in Table 2.

Our experiments consist of two parts. In the first part, our proposed formulations are compared with the full mesh formulation in terms of network installation cost. In the second part, we first measure the maximum possible amount of survivability obtained by full mesh formulation. Indeed, we compute survivability induced by the full mesh formulation between every pair of installed controllers with the aim of a mixed integer programming formulation. Then, the obtained survivability will be given as input to our proposed formulation. It will provide a proper basis of comparison while different formulations are to design low-cost solutions of similar intended survivability requirements. The results of the first experiments are reported in Fig. 1.

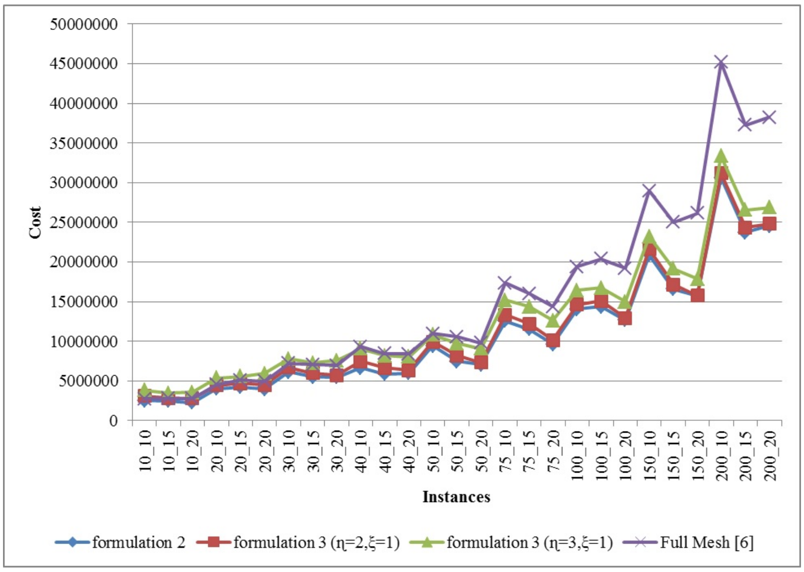

Figure 1: Results of the first part of experiments

In this figure, horizontal axis gives names of the instances tested. In this naming the left-hand digit represents the number of switches used in the instance and the right-hand digit represents the potential controller locations. The vertical axis also gives the best cost found by each of the algorithm. As it can be seen from Fig. 1, networks designed by our formulation (3) with $\zeta_i = 1$, $\forall i \in S$ and $\eta_{pq} = 1, 2, 3$, $\forall pq \in P$ costs much less than the ones found by the full mesh formulation even when higher degrees of survivability is required. One of the main advantages of our proposed model over the existing full mesh formulation is that our proposed model could receive survivability requirement between any pair of controllers, as an input. It will motivate the second part of our experiment. In this experiment, we first use an auxiliary mixed integer programming formulation (describe blew) in order to find the maximum possible edge-disjoint paths between two installed controllers in a network designed by running full mesh formulation. For instance l , we first enumerate the number of installed controllers say n_l. Let K_l be the complete graph of size n_l. For simplicity write $K_l = (N_l, E_l)$. For fixed s and t in N_l.

$$\text{Formulation (4)} : \pi_l^* = \max \quad \pi$$

$$\sum_{ij \in E_l} f_{ij} - \sum_{ji \in E_l} f_{ji} = \begin{cases} \pi & i = s \\ -\pi & i = t \\ 0 & i \neq s,t \end{cases}, \forall i \in N_l \tag{15}$$

$$f_{ij} + f_{ji} \leq 1, \forall ij \in E_l \tag{16}$$

$$f_{ij} \in \{0,1\}, \forall ij \in E_l \tag{17}$$

Since the full mesh formulation will induce a symmetric topology (complete graph) to any pair of installed controllers, the maximum value obtained by running the above formulation is equal for any pair of controllers, in the following experiments, for each instance l we simply set $\eta_{pq} = \pi_l^*$.

The results of these experiments are summarized in Table 3. In this table, for each instance, the following items are reported:

Cost: The cost of best solution found.

imp: Percentage improvement of the proposed formulations over the full mesh formulation as:

$$(18) \qquad \frac{Cost_{mesh} - Cost_{pro}}{Cost_{pro}} * 100$$

In calculating this quantity, $Cost_{mesh}$ and $Cost_{pro}$ functions, give the best cost found by the full mesh formulation and the proposed formulation, respectively.

Table 2: Results of the second part of experiments

instance	Full Mesh			$\eta_{pq} = \pi_l^*$		Proposed vs Full Mesh
$\lvert S\rvert$_$\lvert F\rvert$	Cost	n_l	π_l^*	Cost	n_l	Imp(%)
10_10	2723294	2	1	2538323	5	7.29
10_15	2764740	2	1	2487304	4	11.15
10_20	2802604	2	1	2297930	6	21.96
20_10	4514874	3	2	4452951	4	1.39
20_15	5079927	3	2	4712206	8	7.80
20_20	4972259	2	1	3965589	5	25.39
30_10	7212730	2	2	6691812	6	7.78
30_15	7054907	3	2	5996136	7	17.66
30_20	6943387	3	2	5741447	10	20.93
40_10	9338160	4	3	9089666	6	2.73
40_15	8452059	3	2	6637845	11	27.33
40_20	8380464	3	2	6334901	12	32.29
50_10	10980679	3	2	9942399	8	10.44
50_15	10572863	4	3	9712627	6	8.86
50_20	9794963	3	2	7311682	13	33.96
75_10	17338087	4	3	15199661	8	14.07
75_15	16026016	4	3	14342698	12	11.74
75_20	14384096	4	3	12607629	13	14.09
100_10	19401031	4	3	16410486	9	18.22
100_15	20385784	5	4	18287544	9	11.47
100_20	19178530	4	3	14963087	11	28.17
150_10	28997010	6	5	27114002	8	6.94
150_15	25019616	6	5	23379111	12	7.02
150_20	26125080	6	5	22150666	16	17.94
200_10	45157543	8	7	43797175	8	3.11
200_15	37280604	8	7	36126535	10	3.19
200_20	38212885	8	7	35214644	13	8.51
				Average =		14.13

In Table 3, the first column gives names of the instances. The second to sixth columns report the cost of the best solution found, the number of controllers installed and the degree of survivability by the full-mesh formulation as well as proposed formulation. Finally, the seventh column represents the percentage of improvement of the proposed formulation compared to the full mesh formulation in terms of solution quality. Results reported in Table 3 clearly shows that our proposed formulations are capable of designing much more cost efficient networks, even when higher degrees of network survivability is needed. While full mesh formulation provides no flexibility in selecting the required survivability, our proposed formulations will receive survivability requirement as an input and as a result different amount of survivability could be imposed on different part of the underlying network based on user observations. Other than that, imposing a complete graph over the installed controllers, as every controller needs to get connected to every other controller in a full-mesh topology, is kind of port abuse. Our experiments shows that, imposing such a topology will leave a huge switch to controller connection cost while leaving a small controller to controller connection cost. This indeed will not address a careful trade of between different cost components of the objective function. In comparison, as our results show, the proposed formulations usually find a better trade-off between different cost components of the objective function. Even more, full-mesh formulation could not ensure high degrees of survivability for the cases in which a few number of controllers are installed. As an example when the formulation only installs two controllers, we only have one link in between them and then there is no guarantee ensuring such a link's failure.

5. Conclusion

In this paper, the controller placement problem in SDN has been studied in the presence of survivability requirements. To solve this problem, mathematical formulations are provided. In order to evaluate the performance of the proposed model, experiments have been conducted on several instances of networks. The results obtained from the proposed model are compared with the results of the full mesh model performed on similar instances. Using our proposed formulations while reducing total installation cost, will accept user defined survivability parameter as input and shows superiority over the existing full-mesh formulation when similar survivability requirements are intended.

References

[1] A. Abdelaziz, A. T. Fong, A. Gani, U. Garba, S. Khan, A. Akhunzada, H. Talebian, and K.-K. R. Choo, *Distributed controller clustering in software defined networks*, PloS one, 12 (2017), p. e0174715.

[2] A. Abdi Seyedkolaei, S. A. Hosseini Seno, and A. Moradi, *Dynamic controller placement in software-defined networks for reducing costs and improving survivability*, Transactions on Emerging Telecommunications Technologies, 32 (2021), pp. 2161–3915.

[3] O. Blial, M. Ben Mamoun, and R. Benaini, *An overview on sdn architectures with multiple controllers*, Journal of Computer Networks and Communications, 2016 (2016).

[4] G. Guastaroba, M. Savelsbergh, and M. G. Speranza, *Adaptive kernel search: A heuristic for solving mixed integer linear programs*, European Journal of Operational Research, 263 (2017), pp. 789–804.

[5] Y. Jarraya, T. Madi, and M. Debbabi, *A survey and a layered taxonomy of software-defined networking*, IEEE communications surveys & tutorials, 16 (2014), pp. 1955–1980.

[6] H. K. Khalil, *Nonlinear systems*, Prentice-Hall, Upper Saddle River, NJ, 3 ed., 2002.

[7] S.-C. Lin, P. Wang, and M. Luo, *Control traffic balancing in software defined networks*, Computer Networks, 106 (2016), pp. 260–271.

[8] A. Sallahi and M. St-Hilaire, *Optimal model for the controller placement problem in software defined networks*, IEEE communications letters, 19 (2014), pp. 30–33.

[9] H. Selvi, S. Güner, G. Gür, and F. Alagöz, *The controller placement problem in software defined mobile networks (sdmn)*, Software defined mobile networks (SDMN): beyond LTE network architecture, (2015), pp. 129–147.

[10] A. A. Seyedkolaei, S. A. H. Seno, A. Moradi, and R. Budiarto, *Cost-effective survivable controller placement in software-defined networks*, IEEE Access, 9 (2021), pp. 129130–129140.

[11] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, *Are we ready for sdn? implementation challenges for software-defined networks*, IEEE Communications Magazine, 51 (2013), pp. 36–43.

[12] Y. Wang, Q. Zhong, X. Qiu, and W. Li, *Resource allocation for reliable communication between controllers and switches in sdn*, Journal of Network and Systems Management, 26 (2018), pp. 966–992.

Please cite this article using:

Ahmad Moradi, Ali AbdiSeyedkolaei, Survivable controller placement in software defined network, AUT J. Math. Comput., 3(2) (2022) 185-192
DOI: 10.22060/AJMC.2022.21706.1100