A Heuristic Method for Controller Placement and Enhanced Availability between SDN Controllers

Dorabella Santos
INESC Coimbra
Coimbra, Portugal
dorabella.santos@gmail.com

João P. Vidal
INESC Coimbra
Coimbra, Portugal
joaosantosvidal_1994@hotmail.com

Teresa Gomes, Lúcia Martins
Dep. Electrical and Computer Eng.
University of Coimbra
INESC Coimbra
Coimbra, Portugal
{teresa,lucia}@deec.uc.pt

***Abstract*—The problem of guaranteeing 'five-nines' availability between the controllers in a SDN network is addressed. We consider the problem of controller placement under delay constraints, while assuming a sub-graph of enhanced availability between the controllers. The sub-graph can be upgraded at a given cost, to achieve the desired availability requirements. To solve this problem, we propose a heuristic method to evaluate several controller node sets and the upgrade cost of the sub-graph for enhanced availability. The computational results show that slightly increasing the number of controllers can lead to a significant reduction in the upgrade cost.**

***Index Terms*—SDN, controller placement, availability, Steiner tree, heuristics, integer linear programming**

I. Introduction

In software-defined networking (SDN), the control plane is decoupled from the data plane. The devices in the data plane are basically forwarding devices, leaving the control decisions to the SDN controller in the control plane. Having only one controller in the network, can lead to the network totally crashing if the controller fails. Therefore, several studies have been made on how many controllers to deploy in the network and where to place them. This is known as the controller placement problem (CPP) [1], which has been extensively studied. The set of controllers should function as a logically centralized unit, where each controller has a complete and updated topology view of the complete network. Therefore having too many controllers in the network will increase intercontroller communication overhead [2]. The CPP is mainly driven by delay constraints between the controllers themselves, and between the switches and the controllers that manage them [3], [4].

In this paper, we focus mainly on the control plane availability, namely between the controllers. There has been an increasing number of research on SDN control plane availability. Usually availability is evaluated in terms of control plane connectivity [5], [6] or reachability [7]. In [8], a more exhaustive study was performed to assess the control plane availability against several types of failures. Traditionally, end-to-end availability is increased by path protection [9], which is also considered in the present paper.

This work is funded by ERDF Funds through the Centre's Regional Operational Program and by National Funds through the FCT - Fundação para a Ciência e a Tecnologia, I.P. under the project CENTRO-01-0145-FEDER-029312. This work was also partially supported by FCT under project grant UIDB/00308/2020.

Controller placement can be used to increase control plane availability. In [10], the CPP variant with backup controllers is studied in order to guarantee the 'five nines' requirement (i.e. an availability of at least 0.99999) for the control plane. In [11], the availability is evaluated in terms of the expected percentage of control path loss (a control path is the path connecting a switch to its controller). The authors propose several heuristics for the controller placement, in order to increase of the availability of the control paths. In [12], a response time and availability study is conducted for the distributed SDN control plane.

The desired availability is not always achievable by repositioning the controllers and/or by guaranteeing path protection [13]. To ensure high availability in the control plane between the controllers, we assume a sub-graph whose links can be upgraded to have higher availability, for example by reducing the average time to repair on the links. We focus on the intercontroller connections, since the set of controllers are the brain of the SDN network. Although not as frequent as switch-controller communication, intercontroller communication is critical for the control and management of the network [14].

In this work, we assume that there is a sub-graph connecting all the controllers, and that the links of the sub-graph can be upgraded to have enhanced availability at a given cost [15]. This is based on the spine concept first proposed in [16] and explored in [13]. Related previous works [17] and [18] also address the CPP and spine link upgrade problem. However, both these works consider availability guarantees between the switches and the controllers that manage them, and not between the controllers themselves. While path protection is not considered in [17], it is considered in [18].

The innovation of this paper is incorporating the high-availability spine concept as a Steiner tree structure, where the controller nodes are the terminal nodes. The primary paths between the controllers are routed on the Steiner tree, whereas the backup paths are node-disjoint to the respective primary paths and are not imposed to be routed on the Steiner tree. The spine problem alone of selecting which links to upgrade is NP-complete [15] and the Steiner tree problem is also

NP-complete [19]. Therefore, our joint optimization problem is NP-complete. Hence, we propose a heuristic method to solve the problem of placing the controllers driven by delay requirements, and of determining a Steiner tree substructure for enhanced availability driven by the controller placement and the desired availability requirements.

The paper is organized as follows: in Section II we present the controller placement and Steiner tree upgrade problem; in Section III we describe the heuristic method implemented to solve the problem; in Section IV we analyze the computational results obtained; and in Section V we draw some conclusions and propose some approaches for future work.

II. Controller Placement and Steiner Tree Upgrade

The optimization problem addressed in this paper deals with several issues: (i) controller placement satisfying delay constraints; (ii) determination of a Steiner tree connecting all the controllers, to have enhanced availability; (iii) determination of protection routing between controllers; (iv) cost of the availability enhancement of the Steiner tree, to achieve the availability requirements between controllers. This problem is inherently multiobjective. In general, finding a Steiner tree using integer linear programming (ILP) can be very time consuming. Therefore, solving our joint optimization problem exactly is impractical. We break down the optimization problem into subproblems and we choose a heuristic approach to tackle it.

A. Controller Placement Problem

Consider that the SDN data plane is represented by a graph $G = (N, E)$, where N is the set of nodes and E is the set of links. Each link is represented by its end nodes $\{i, j\}$. We assume the delay between two nodes is proportional to the shortest path length between them and is denoted as d_{ij} [1].

We assume that the delay between each switch and the controller that manages it cannot exceed a given maximum value D_{sc}, while the delay between any two controllers cannot exceed a given maximum value D_{cc}. Since communication between the controllers and the switches they manage is more frequent than the intercontroller communciation, we assume that $D_{sc} < D_{cc}$ [3].

The number of controllers and their placements are determined by an ILP model, satisfying the delay constraints and minimizing the number of controllers. The objective function is motivated by the fact that intercontroller communication overhead should be kept to a minimum, so not to jeopardize the network's performance. Consider the following decision variables:

- $y_i \in \{0, 1\}$ binary variable that is 1 if there is a controller placed in node i, and 0 otherwise

The ILP model for the controller placement is given by

$$\min \sum_{i \in N} y_i \tag{1}$$

s.t.

$$\sum_{\substack{j \in N: \\ d_{ij} \leq D_{sc}}} y_j \geq 1 \quad i \in N \tag{2}$$

$$y_i + y_j \leq 1 \qquad i, j \in N : d_{ij} > D_{cc} \tag{3}$$

$$y_i \in \{0, 1\} \qquad i \in N \tag{4}$$

The objective function (1) aims to minimize the number of controllers. Constraints (2) guarantee that for any node i, there is a controller distanced at most D_{sc} from it. Constraints (3) guarantee that two controller nodes cannot be distanced further than D_{cc}. Finally, constraints (4) are the variable domain constraints.

B. Steiner Tree as the Control Plane Sub-graph

Once the controller placement is known, it is possible to determine a sub-graph connecting all the controllers. This sub-graph or spine, should be designed to provide highly available intercontroller connections in the control plane. In this paper, we have assumed that the sub-graph is a Steiner tree, where the controller nodes are the terminal nodes.

The primary paths connecting any two controllers are routed over the Steiner tree, and should satisfy the maximum intercontroller delay value D_{cc}. To increase intercontroller availability, backup protection is considered. Therefore, for any two controllers, a node-disjoint backup path is also computed (which is not imposed to avoid using the Steiner tree).

We assume that the end-to-end availability of the intercontroller connections must be at least a given minimum value Λ. This cannot always be achieved by backup protection alone. Therefore, we consider that the links belonging to the Steiner tree can be upgraded to have enhanced availability at a given cost.

We consider that each link of the network has a default distance-based availability given as in [20] (page 186):

$$a_{ij}^0 = 1 - \frac{MTTR}{MTBF_{ij}} \tag{5}$$

where $MTTR$ denotes the mean time to repair (in hours) and $MTBF_{ij}$ denotes the mean time between failures (in hours) of link $\{i, j\}$. We assume $MTTR = 24$ h and $MTBF_{ij} = CC \times 365 \times 24/\ell_{ij}$ where $CC = 450$ km results from the cable cut rate and ℓ_{ij} denotes the link length.

Each link of the Steiner tree can be upgraded to different levels of enhanced availability. In each level, the link unavailability is decreased by a given value $\varepsilon \in (0, 1)$. Assuming K levels, the unavailability of link $\{i, j\}$ upgraded to level k is given by $u_{ij}^k = (1 - \varepsilon) u_{ij}^{k-1}$, $k = 1, ..., K$ where $u_{ij}^0 = 1 - a_{ij}^0$ denotes the default unavailability (we do not consider any downgrade level here) [15]. In terms of availability, we have that $u_{ij}^k = 1 - a_{ij}^k$ and so $a_{ij}^k = a_{ij}^{k-1} + \varepsilon(1 - a_{ij}^{k-1})$, $k = 1, ..., K$.

The cost for upgrading the link availability to level k is given by [15], [21],

$$c_{ij}^k = -\ell_{ij} \cdot \ln\left(\frac{1 - a_{ij}^k}{1 - a_{ij}^0}\right) \quad k = 1, ..., K \tag{6}$$

C. Steiner Tree Upgrade Problem

The problem of upgrading the Steiner tree links can be formulated as an ILP model. Consider set $\mathcal{CP}$ as the set of controller nodes given by (1)-(4), i.e., the set of nodes i such that $y_i = 1$ in the solution. Moreover, consider set $\mathcal{ST}$ as the set of links belonging to the Steiner tree selected for interconnecting the controller nodes. Consider $p_{c_1 c_2}$ and $b_{c_1 c_2}$ as the primary and backup paths between controller nodes c_1 and c_2, respectively.

Since only the primary paths are imposed to be routed on the Steiner tree, they are the ones that can impose the necessary link upgrade that is needed to satisfy the minimum value Λ for the end-to-end availability between each pair of controllers. Hence for each pair of controller nodes c_1 and c_2, the default backup path availability $\mathcal{A}^{b0}_{c_1 c_2}$ is computed (as the product of the default availabilities of the links belonging to the path). Then, the necessary primary path availability $\mathcal{A}^{p}_{c_1 c_2}$ can also be computed to satisfy $1-(1-\mathcal{A}^{p}_{c_1 c_2})(1-\mathcal{A}^{b0}_{c_1 c_2}) = \Lambda$.

Consider the following decision variables:

- $z^0_{ij} \in \{0,1\}$ binary variable that is 1 if link $\{i,j\} \in \mathcal{ST}$ is not upgraded, and 0 otherwise;
- $z^k_{ij} \in \{0,1\}$ binary variable that is 1 if link $\{i,j\} \in \mathcal{ST}$ is upgraded to level k, and 0 otherwise ($k = 1, ..., K$).

The ILP model for upgrading the links of the Steiner tree is given by

$$\min \sum_{k=1}^{K} \sum_{\{i,j\} \in \mathcal{ST}} c^k_{ij} z^k_{ij} \tag{7}$$

s.t.

$$\sum_{k=0}^{K} z^k_{ij} = 1 \qquad \{i,j\} \in \mathcal{ST} \tag{8}$$

$$\sum_{(i,j) \in p_{c_1 c_2}} \sum_{k=0}^{K} z^k_{ij} \log(a^k_{ij}) \geq \log\left(\mathcal{A}^{p}_{c_1 c_2}\right) \quad c_1, c_2 \in \mathcal{CP} \tag{9}$$

$$z^k_{ij} \in \{0,1\} \qquad \{i,j\} \in \mathcal{ST},\ k = 0, ..., K \tag{10}$$

The objective function (7) is the minimization of the cost of upgrading the links of the Steiner tree to level $k \geq 1$. Constraints (8) ensure that each link is either not upgraded or is upgraded to only one level $k \geq 1$ simultaneously. Constraints (10) are the variable domain constraints.

Constraints (9) ensure that the availability of each primary path must be at least the necessary availability $\mathcal{A}^{p}_{c_1 c_2}$. Since these constraints are nonlinear in nature, they have been linearized in the classical manner. Recall that the default availability of a path p is given by

$$\mathcal{A}^0 = \prod_{\{i,j\} \in p} a^0_{ij} \tag{11}$$

Applying the logarithm to (11), we have that

$$\log(\mathcal{A}^0) = \sum_{\{i,j\} \in p} \log(a^0_{ij}) \tag{12}$$

Since each link is either not upgraded or upgraded to a given level k, and since variables z^k_{ij} carry this information, the logarithm of the upgraded availability is given by

$$\log(\mathcal{A}) = \sum_{\{i,j\} \in p} \sum_{k=0}^{K} z^k_{ij} \log(a^k_{ij}) \tag{13}$$

which is the expression appearing in constraints (9). A more general and detailed derivation can be found in [18].

III. Heuristic Method

Different controller placements provide different Steiner trees, and in turn different upgrading costs. Finding optimal Steiner trees with delay constraints using exact methods can be time-consuming [22]. The problem addressed here involves determining the controller placement set and then obtaining a Steiner tree between the controllers. Finally, the links of the Steiner tree for upgrading are selected. Therefore, as the overall problem is highly complex, we implemented a first-approach heuristic to evaluate different controller placements and the upgrade costs of the resulting Steiner trees.

The pseudo-code for the heuristic is shown in Fig. 1. The heuristic starts by solving the ILP model given by (1)-(4) for the CPP (line 5). The solution provides the first set of controller nodes $\mathcal{CP}$, or the problem is infeasible and so there is no controller placement that can satisfy the delay constraints and the heuristic stops (lines 6-7).

In case a controller placement set $\mathcal{CP}$ exists, then a Steiner tree $\mathcal{ST}$ is determined by using Takahasi's algorithm [23]. The algorithm is based on the link lengths and considers the controller nodes in $\mathcal{CP}$ as the terminal nodes of the Steiner tree (line 9).

For each pair of controllers (lines 10-15), the primary path over the Steiner tree is determined (line 11). Then a shortest node-disjoint backup path is computed (line 12) and its default availability is determined (line 13). Finally, the primary path availability which is necessary to satisfy the availability minimum Λ is computed (line 14).

Sometimes, the heuristic cannot find a Steiner tree, such that the primary paths routed over it, satisfies the maximum delay value D_{cc} between controllers. In these cases, the controller placement set is considered as an invalid solution (in the heuristic method), and so is not included in the set of obtained solutions, $Sols$ (line 18). This is omitted in Fig. 1 for readability. We do not consider that the backup paths need to satisfy the maximum delay value D_{cc}.

Once all the required primary path availabilities have been computed, the ILP model given by (7)-(10), is solved for the Steiner tree upgrade problem (line 16). The solution provides the upgrade cost, $cost$, and the level of upgrade of the set of links, $\mathcal{L}$. If the problem is infeasible, then the upgrade problem cannot achieve the required primary availabilities, and the set of controller nodes is considered as an invalid set in the heuristic. If the set of controller nodes is a valid set, then $\mathcal{CP}$ is added to the solution set, $Sols$ (line 18). If the upgrade cost is zero (line 19), then the heuristic also stops

```
1: $\mathcal{VCP} \leftarrow \{\}$    ▷ Empty set of visited CPP sets
2: $Sols \leftarrow \{\}$    ▷ Empty set of valid CPP solutions
3: $continue \leftarrow true$
4: while $continue = true$ do
5:   $\mathcal{CP} \leftarrow ControllerPlacement(D_{sc}, D_{cc}) \backslash \mathcal{VCP}$
6:   if $\mathcal{CP} = \{\}$ then
7:     $continue \leftarrow false$
8:   else
9:     $\mathcal{ST} \leftarrow TakahasiAlgorithm(\mathcal{CP})$
10:    for all $c_1, c_2 \in \mathcal{CP}$ do
11:      $p_{c_1 c_2} \leftarrow$ primary path on $\mathcal{ST}$ between $c_1$ and $c_2$
12:      $b_{c_1 c_2} \leftarrow$ node-disjoint backup path
13:      $\mathcal{A}^{b0}_{c_1 c_2} \leftarrow$ default availability of $b_{c_1 c_2}$
14:      $\mathcal{A}^{p}_{c_1 c_2} \leftarrow (\Lambda - \mathcal{A}^{b0}_{c_1 c_2})/(1 - \mathcal{A}^{b0}_{c_1, c_2})$
15:    end for
16:    $(cost, \mathcal{L}) \leftarrow SteinerTreeUpgrade(\{\mathcal{A}^{p}_{c_1 c_2}\})$
17:    if $\mathcal{L} \neq \{\}$ then
18:      $Sols \leftarrow Sols \cup \{\mathcal{CP}\}$
19:      if $cost = 0$ then
20:        $continue \leftarrow false$
21:      end if
22:    end if
23:    $\mathcal{VCP} \leftarrow \mathcal{VCP} \cup \{\mathcal{CP}\}$
24:  end if
25: end while
```

Fig. 1. Pseudo-code for the heuristic method proposed.

since the upgrade cost can improve no further (no upgrading is needed).

The controller set $\mathcal{CP}$ is also added to the visited controller sets $\mathcal{VCP}$ (line 23), which is then used to eliminate the already obtained sets from the search space of the ILP model for the CPP. Another set of controller nodes is then computed by the ILP model given by (1)-(4), by adding a constraint to eliminate the previous sets from the search space (line 5). Assume that the current number of controllers is C. Then if $\gamma_1, ..., \gamma_C$ denotes the set of controller nodes obtained, the constraint to eliminate this set from the search space is given by $y_{\gamma_1} + \cdots + y_{\gamma_C} \leq C - 1$

However, if a set of C controller nodes is no longer possible, the ILP will return a set of $C' > C$ of controller nodes. Then all the sets of η controller nodes $\gamma_1, ..., \gamma_\eta$, such that $\eta < C'$, are eliminated by the set of constraints

$$\sum_{i \in N} y_i \geq C' \tag{14}$$

$$y_{\gamma_1} + \cdots + y_{\gamma_\eta} \leq C' - 1 \tag{15}$$

and the sets of C' controller sets previously obtained $\gamma_1, ..., \gamma'_C$ are eliminated by the constraints $y_{\gamma_1} + \cdots + y_{\gamma'_C} \leq C' - 1$.

When the ILP model given by (1)-(4) eventually becomes infeasible, then all the possible controller sets have been found and the heuristic stops (line 6-7), else the heuristic proceeds (lines 9-23).

It may happen that there is a large number of possible controller sets, and so another stopping criteria is that if the upgrade cost, *cost*, is higher than zero but does not decrease after a given maximum number of valid solutions, the heuristic stops. This is also omitted in Fig. 1 for readability.

In the end, we have the set of obtained valid solutions in $Sols$. We then evaluate these solutions in terms of the trade-off between the number of controllers and upgrade cost. We consider the non-dominated solutions in function of these two objectives: minimizing the number of controllers and minimizing the upgrade cost. A non-dominated solution is such that any other solution which has a smaller number of controllers must have a higher upgrade cost, or that has a smaller upgrade cost must have a higher number of controllers. In other words, a non-dominated solution is such that it is not possible to further improve both objectives simultaneously.

In the next section, we will loosely use the term *non-dominated* solutions, for the solutions which are not dominated by any solution in set *Sols*, although these solutions might be dominated by other solutions not found by the heuristic.

IV. Computational Results

To evaluate our heuristic we have used the following networks: polska, nobel_germany and cost266 networks from SNDlib [24], and the spain network from [25]. The topological characteristics of the networks are summarized in Table I, which shows the number of nodes, the number of edges, the average node degree and the graph diameter (longest shortest path between any two nodes) for each network.

Network	#nodes	#links	avg deg	D_g [km]
polska	12	18	3.00	811
spain	14	22	3.14	1034
nobel_germany	17	26	3.06	790
cost266	37	57	3.08	4032

TABLE I
Topological characteristics of the networks

We have considered the maximum delay values D_{sc} and D_{cc} given as percentages of the graph diameter D_g [3]. We have considered the D_{sc} values to be 35%, 40%, 45%, while the D_{cc} values were considered to be 65%, 70%, 75%. For all the networks, we have considered the minimum availability between controllers to be ‘five-nines’, $\Lambda = 0.99999$. To obtain the desired availability levels, each link of the Steiner tree can be upgraded up to $K = 4$ levels where, for each level, the link unavailability is reduced by a factor of $\varepsilon = 0.5$.

Next we will discuss in detail the results obtained for the networks in Table I, which will show that in most instances path protection is insufficient to ensure the target ‘five nines’ availability. Results will also be discussed seeking to highlight the trade-off between the number of controllers and the cost of upgrading the links to achieve the desired availability.

For the nobel_germany network, in all instances the minimum number of controllers was 2, and the best sets of 2 controllers do not need upgrading any of the links to achieve the desired availability. This means that for nobel_germany, path protection is sufficient to ensure the availability requirements.

In turn, the results for the polska network are shown in Table II, while for the spain network are shown in Table III, and for the cost266 network are shown in Table IV. The tables show the respective D_{sc} and D_{cc} values for each instance. In column 'C' the number of controllers is shown for each non-dominated solution and the respective upgrade cost in column 'cost'. If '-' is shown in these columns, the instance is either infeasible if a set of controllers cannot be obtained for the given maximum delay values, or the heuristic failed to find a valid Steiner tree for each set of controllers obtained, using Takahashi's algorithm, although one may exist. A valid Steiner tree is, in this sense, a Steiner tree that satisfies the maximum D_{cc} delay between any pair of controllers on the Steiner tree.

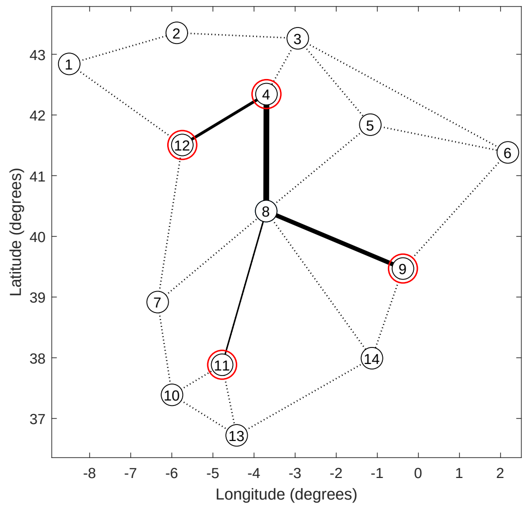
In column '#upg', the total number of upgraded links is shown, and in the group of columns 'k' the number of links for each level of upgrade $k = 1, 2, 3, 4$ is also shown. For each non-dominated solution, the minimum, average and maximum end-to-end availabilities are shown as their corresponding unavailabilities, in columns '$1 - \min \mathcal{A}$', '$1 - \text{avg}\mathcal{A}$' and $1 - \max \mathcal{A}$' respectively. The last column 't(s)' refers to the total runtime for the heuristic in each instance.

As can be seen in Table II, for all instances there are sets with 3 controllers where no upgrading is necessary. For $D_{sc} = 35\%$ and $D_{sc} = 40\%$, 3 controllers is the minimum number. However, for $D_{sc} = 45\%$, the minimum number is 2 controllers, for which the upgrade cost is 274.9. In this case, 2 links are upgraded to level $k = 1$. Note that for $D_{sc} = 35\%$ and for $D_{cc} = 65\%$ and 70%, the heuristic could not obtain a solution. In these instances, the heuristic failed to find a valid Steiner tree.

As D_{sc} and D_{cc} increases, the number of possible controller placement sets also increases, while the minimum number of required controllers decreases. This results in improvement of the upgrade cost for a fixed number of controllers. This effect is better observed in Tables III and IV.

As can be seen in Table III, only for $D_{sc} = 45\%$, having 3 controllers yields solutions with upgrade cost equal to zero (best possible solution). For smaller D_{sc} values, either the instance is infeasible when $D_{cc} = 65\%$, or the smallest possible number of controllers is 4.

The best controller set found for $C = 5$ has an upgrade cost of 850.5, where 2 links are upgraded to level $k = 1$ and 1 link is upgraded to level $k = 3$. The best controller set for $C = 4$ has an higher upgrade cost. This is illustrated in Fig. 2 for $D_{sc} = 40\%$ and $D_{cc} = 70\%$, where the Steiner tree for each case is shown as solid lines, where the thickness of each link is proportional to the level of upgrade. The controller nodes are marked with red circles. For 4 controllers (top network), node 8 is Steiner node, link {8,11} is not upgraded, link {4,12} is upgraded to level $k = 1$, link {8,9} is upgraded to level $k = 2$ and the link {4,8} is upgraded to level $k = 3$. For 5 controllers (bottom network), there are no Steiner nodes (just terminal nodes) and link {8,7} is not upgraded, links {4,8} and {4,12} are upgraded to level $k = 1$, and link {5,8} is upgraded to level $k = 3$. Although in both cases a total of 3 links are upgraded, the cost with 5 controllers is lower because 2 links are now upgraded to level $k = 1$, instead of having one of these upgraded to level $k = 2$. Recall that according to the cost function used in (6), upgrading to a higher level causes the cost to grow exponentially.



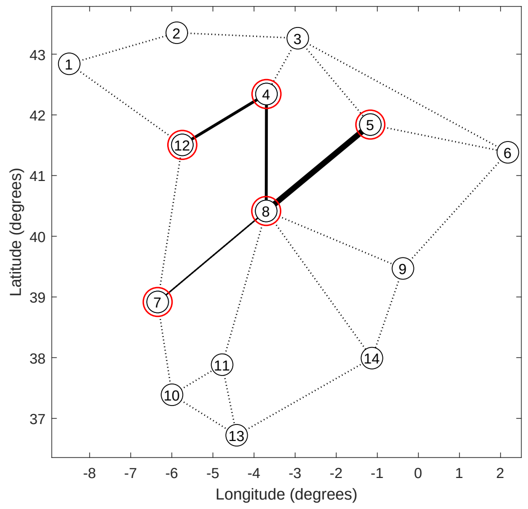


Fig. 2. Spain network for $D_{sc} = 40\%$ and $D_{cc} = 70\%$, with 4 controllers (top) and 5 controllers (bottom). The controller nodes are marked with red circles, and the Steiner tree is shown in solid lines, where the thickness of the solid lines reflects the upgrade level of the links

Adding more than 5 controllers does not yield better costs – in other words, solutions with a higher number of controllers are dominated by the solution with 5 controllers. As expected, relaxing the D_{sc} delay values can improve the values of the upgrade cost. Note that when D_{sc} is relaxed from 35% to 40%, the upgrade cost for $C = 4$ decreases from 1212.3

D_{sc}	D_{cc}	C	cost	#upg	k				$1-\min\mathcal{A}$	$1-\text{avg}\mathcal{A}$	$1-\max\mathcal{A}$	t(s)
					1	2	3	4				
35%	65%	-	-	-	-	-	-	-	-	-	-	-
	70%	-	-	-	-	-	-	-	-	-	-	-
	75%	3	0	0	0	0	0	0	7E-06	6E-06	4E-06	2.36
40%	65%	3	0	0	0	0	0	0	1E-05	9E-06	8E-06	0.24
	70%	3	0	0	0	0	0	0	1E-05	9E-06	8E-06	0.57
	75%	3	0	0	0	0	0	0	8E-06	6E-06	3E-06	0.76
45%	65%	3	0	0	0	0	0	0	8E-06	5E-06	2E-06	0.66
	70%	2	274.9	2	2	0	0	0	1E-05	1E-05	1E-05	1.08
		3	0	0	0	0	0	0	1E-05	9E-06	8E-06	
	75%	2	274.9	2	2	0	0	0	1E-05	1E-05	1E-05	1.49
		3	0	0	0	0	0	0	9E-06	6E-06	3E-06	

TABLE II
NON-DOMINATED SOLUTIONS FOR POLSKA WITH $\Lambda = 0.99999$

D_{sc}	D_{cc}	C	cost	#upg	k				$1-\min\mathcal{A}$	$1-\text{avg}\mathcal{A}$	$1-\max\mathcal{A}$	t(s)
					1	2	3	4				
35%	65%	-	-	-	-	-	-	-	-	-	-	-
	70%	4	1212.3	5	3	1	1	0	9E-06	7E-06	3E-06	3.75
		5	850.5	3	2	0	1	0	9E-06	5E-06	1E-06	
	75%	4	1212.3	5	3	1	1	0	9E-06	7E-06	3E-06	7.68
		5	850.5	3	2	0	1	0	9E-06	5E-06	1E-06	
40%	65%	-	-	-	-	-	-	-	-	-	-	-
	70%	4	999.5	3	1	1	1	0	9E-06	7E-06	2E-06	4.98
		5	850.5	3	2	0	1	0	9E-06	5E-06	1E-06	
	75%	4	999.5	3	1	1	1	0	9E-06	7E-06	2E-06	7.99
		5	850.5	3	2	0	1	0	9E-06	5E-06	1E-06	
45%	65%	3	0	0	0	0	0	0	6E-06	6E-06	6E-06	1.64
	70%	3	0	0	0	0	0	0	6E-06	6E-06	6E-06	3.31
	75%	3	0	0	0	0	0	0	6E-06	6E-06	6E-06	3.51

TABLE III
NON-DOMINATED SOLUTIONS FOR SPAIN WITH $\Lambda = 0.99999$

to 999.5. However, for these instances relaxing the D_{cc} values does not yield improvements in cost upgrade, since $D_{sc} \ll D_{cc}$. Moreover, relaxing the delay values also results in that the minimum number of controllers required decreases (minimum of 4 controllers for $D_{sc} = 35\%$ and 40%, while for $D_{sc} = 45\%$ the minimum number possible number of controllers is 3). The runtimes are still quite small for all instances in this network.

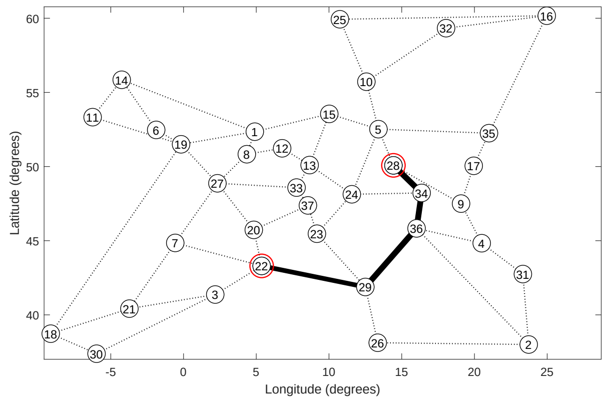
As can be seen in Table IV for the cost266 network, for $D_{sc} = 35\%$ and for $D_{cc} = 65\%$, there are two non-dominated solutions, one with 3 controllers and another with 4 controllers. With the increase of D_{cc} only one of these two solutions can be found. This is because the number of possible solutions increases and the search stops before the best 4 controller solution is obtained. For $D_{sc} = 40\%$, note that increasing the number of controllers from 2 to 3, the upgrade cost improves although the number of upgraded links increases from 4 to 6. This is illustrated in Fig. 3. In the case for 2 controllers, link $\{22, 29\}$ is upgraded to level $k = 3$, while the remaining 3 links are upgraded to level $k = 4$. In the case for 3 controllers, links $\{1, 15\}$ and $\{5, 15\}$ are upgraded to level $k = 4$, while the remaining 4 links are upgraded to level $k = 3$. The sum of the link lengths upgraded to level $k = 3$ is 604.07 and 1111.52 for 2 and 3 controllers, respectively. We can observe that the sum is higher for 3 controllers. However, the sum of the link lengths upgraded to level $k = 4$ is 1038.73 and 620.89 for 2 and 3 controllers, respectively. Since the sum of the link lengths for $k = 4$ is much higher for 2 controllers, this impacts the upgrade cost more significantly than the links upgraded to $k = 3$, resulting in a reduction of the cost for 3 controllers.

A related observation can be made for $D_{sc} = 45\%$, where the upgrade cost for 3 controllers is better than that for 2 controllers, although the number of upgraded links are the same. In both cases, 2 links are upgraded to level $k = 2$, and 1 link is upgraded to level $k = 3$. The sum of the lengths of the two links upgraded to level $k = 2$ is slightly smaller for the case with 2 controllers than for 3 controllers. However, the link upgraded to level $k = 3$ is longer for the case with 2 controllers, than for the case with 3 controllers, and according to (6) the length of this link has a higher impact on the upgrade cost. Therefore, there is an improvement of the cost for the case with 3 controllers. Increasing the number of controllers to 4, the best solution obtained has a cost of 1855.38 with 5 links upgraded to level $k = 2$, which is clearly dominated by the previous solutions.

Note that the spain network is smaller than the cost266 network, as shown by the graph diameter in Table I. The spain network requires more controllers than cost266, since the percentages used reflect a much smaller absolute distance for spain than for cost266. The runtimes for cost266 are significantly higher, although still reasonable.

D_{sc}	D_{cc}	C	cost	#upg	k				$1-\min\mathcal{A}$	$1-\text{avg}\mathcal{A}$	$1-\max\mathcal{A}$	t(s)
					1	2	3	4				
35%	65%	3	5658.5	7	0	0	1	6	1E-05	7E-06	1E-06	87.72
		4	5317.3	7	0	0	3	4	1E-05	5E-06	1E-06	
	70%	3	5658.5	7	0	0	1	6	1E-05	7E-06	1E-06	70.36
	75%	3	5658.5	7	0	0	1	6	1E-05	7E-06	1E-06	107.20
40%	65%	2	4136.1	4	0	0	1	3	1E-05	1E-05	1E-05	28.12
		3	4032.8	6	0	0	4	2	1E-05	6E-06	2E-06	
	70%	2	4136.1	4	0	0	1	3	1E-05	1E-05	1E-05	37.95
		3	4032.8	6	0	0	4	2	1E-05	6E-06	2E-06	
	75%	2	4136.1	4	0	0	1	3	1E-05	1E-05	1E-05	49.48
		3	4032.8	6	0	0	4	2	1E-05	6E-06	2E-06	
45%	65%	2	1402.7	3	0	2	1	0	9E-06	9E-06	9E-06	24.57
	70%	2	1402.7	3	0	2	1	0	9E-06	9E-06	9E-06	36.84
		3	1352.4	3	0	2	1	0	1E-05	7E-06	5E-06	
	75%	2	1402.7	3	0	2	1	0	9E-06	9E-06	9E-06	66.66
		3	1352.4	3	0	2	1	0	1E-05	7E-06	5E-06	

TABLE IV
NON-DOMINATED SOLUTIONS FOR COST266 WITH $\Lambda = 0.99999$



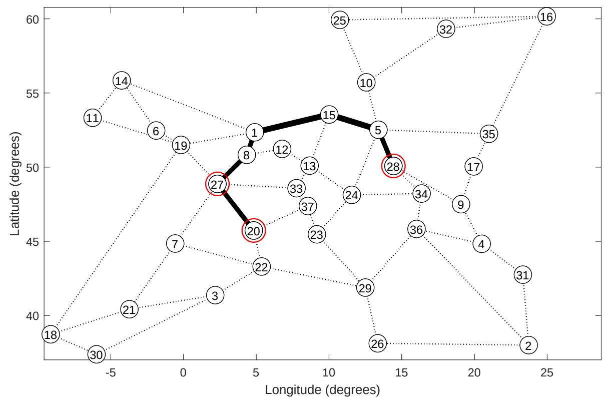


Fig. 3. Cost266 network for $D_{sc} = 40\%$ and $D_{cc} = 70\%$, with 2 controllers (top) and 3 controllers (bottom). The controller nodes are marked with red circles, and the Steiner tree is shown in solid lines, where the thickness of the solid lines reflects the upgrade level of the links.

In all tables, the minimum, average and maximum end-to-end availabilities are shown, translated into unavailabilities for better readability. Analyzing the second last column of Table II, i.e. column '$1 - \max\mathcal{A}$', we can see that for $D_{cc} = 75\%$ and 3 controllers, although there is no link upgrade, there is at least one pair of paths that has an excess end-to-end availability, i.e., much higher than 0.99999. For the solution with 2 controllers, the availabilities achieved by upgrading the 2 links is just enough to satisfy the target availability.

In Table III, we can see that all feasible instances present excess end-to-end availability for one or more path pairs, although the cases without upgrade (with zero cost) present the smallest excess. In Table IV, we can also observe excess availability for $C \geq 3$. For 2 controllers, the target availabilities achieved are at the limit. These observations means that a more careful analysis may reveal that, since some backup paths use upgraded links of the Steiner tree, the links being used by the respective primary paths may actually be downgraded (from a higher level to a lower level). In other words, there may be room for a post-processing optimization for the upgrade cost of the Steiner tree, in future work.

V. CONCLUSIONS

We have addressed the problem of controller placement and link availability upgrade of a Steiner tree sub-graph connecting the controllers. This multiobjective optimization problem is NP-complete. Solving it using exact methods is impractical due to its high complexity, and so we propose a heuristic approach, which tackles the problem broken down into its constituent subproblems. The first subproblem is the controller placement problem under delay constraints which is solved via an ILP model. The heuristic starts by determining sets of controller nodes, starting with the minimum possible number for the maximum D_{sc} and D_{cc} values imposed. The different controller sets, which are given as input to Takahashi's algorithm, lead to different Steiner trees connecting the controllers. The upgrade cost of the links belonging to the Steiner tree is computed to achieve 'five-nines' availability between the controllers and solved via an ILP model. The solutions are then evaluated considering the number of controllers against the upgrade cost. The non-dominated solutions in the set are then retrieved. Since this is a heuristic, these non-dominated solutions may not be *truly* *non-dominated*, since better solutions may exist although the heuristic did not find them.

It is possible to conclude that in general, increasing the number of controllers from the minimum number to one more, usually translates in a drastic cost reduction. However, adding more controllers does not usually improve the cost significantly or at all. This was also observed in the related work [18]. However the problem dealt in this reference, is much simpler because the spine is considered as a spanning tree to guarantee the desired end-to-end availability between the switches and the controllers managing each one. The fact that here we consider only the connections between the controllers, and therefore, the spine is now a Steiner tree, significantly increases the complexity of the problem (the Steiner problem is NP-complete).

Due to control plane performance, the number of controllers in the network should not be too large. Therefore, having a little more than the required minimum is usually sufficient to get good end-to-end availabilities, given the placements for the best sets. It was also observed that the links upgraded to higher availability levels have a much more significant impact on the cost, as is expected by the cost function used. This is particularly noticeable in the computational results of cost266, where having a total of 4 upgraded links – 3 links upgraded to level $k = 4$ and 1 link upgraded to level $k = 3$ – is more costly than having a total of 6 upgraded links – 4 links upgraded to level $k = 3$ and 2 links upgraded to level $k = 4$.

The upgraded links prove, that the required availability guarantees cannot be achieved by path protection alone. This means that without the spine sub-graph, the required end-to-end availabilities could not be achieved as pointed out in [15], except for those instances where upgrading was not necessary. For the instances with zero upgrade cost, path protection is sufficient to guarantee the desired availabilities (as was observed for nobel_germany).

Finally, we observed excess end-to-end availabilities in many instances. This means that there may be links that can be upgraded to lower levels, if some of the backup paths happen to use any of the upgraded links. Therefore, in future work, an analysis of the excess end-to-end availability in some of the paths, will be carried out. This analysis may allow for post-processing optimization of the upgrade cost of the Steiner tree, by downgrading some of the upgraded links to lower levels, consequently reducing the cost.

Acknowledgment

This work was partially supported by the COST Action CA15127 (“Resilient communication services protecting end user applications from disaster-based failures – RECODIS”).

References

[1] B. Heller, R. Sherwood, and N. McKeown, “The controller placement problem,” in *ACM HotSDN*, New York, USA, 2012, pp. 7–12.

[2] A. S. Muqaddas, P. Giaccone, A. Bianco, and G. Maier, “Inter-controller traffic to support consistency in ONOS clusters,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 1018–1031, 2017.

[3] N. Perrot and T. Reynaud, “Optimal placement of controllers in a resilient SDN architecture,” in *DRCN*, Paris, France, 2016, pp. 145–151.

[4] D. Santos, A. de Sousa, and C. M. Machuca, “Robust SDN controller placement to malicious node attacks,” in *DRCN 2018, co-located with ICIN 2018*, Paris, France, 2018, pp. 1–8.

[5] G. Nencioni, B. E. Helvik, A. J. Gonzalez, P. E. Heegaard, and A. Kamisinski, “Availability modelling of software-defined backbone networks,” in *DSN-W*, Toulouse, France, 2016, pp. 105–112.

[6] G. Nencioni, B. E. Helvik, and P. E. Heegaard, “Including failure correlation in availability modeling of a software-defined backbone network,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 1032–1045, 2017.

[7] T. A. Nguyen, T. Eom, S. An, J. S. Park, J. B. Hong, and D. S. Kim, “Availability modeling and analysis for software defined networks,” in *PRDC*, Zhangjiajie, China, 2015, pp. 159–168.

[8] P. Vizarreta, P. Heegaard, B. Helvik, W. Kellerer, and C. M. Machuca, “Characterization of failure dynamics in SDN controllers,” in *RNDM*, Alghero, Italy, 2017, pp. 1–7.

[9] S. Song, H. Park, B. Choi, T. Choi, and H. Zhu, “Control path management framework for enhancing software-defined network (SDN) reliability,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 2, pp. 302–316, 2017.

[10] F. J. Ros and P. M. Ruiz, “Five nines of southbound reliability in software-defined networks,” in *ACM HotSDN*, New York, USA, 2014, pp. 31–36.

[11] Y. Hu, W. Wendon, X. Gong, X. Que, and C. Shiduan, “Reliability-aware controller placement for software-defined networks,” in *IM*, Ghent, Belgium, 2013, pp. 672–675.

[12] E. Sakic and W. Kellerer, “Response time and availability study of raft consensus in distributed sdn control plane,” *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 304–318, 2018.

[13] A. Alashaikh, T. Gomes, and D. Tipper, “The spine concept for improving network availability,” *Computer Networks*, vol. 82, pp. 4 – 19, 2015.

[14] T. Zhang, A. Bianco, and P. Giaccone, “The role of inter-controller traffic in SDN controllers placement,” in *NFV-SDN*, Palo Alto, USA, 2016, pp. 87–92.

[15] A. Alashaikh, D. Tipper, and T. Gomes, “Embedded network design to support availability differentiation,” *Annals of Telecommunications*, vol. 74, no. 9-10, pp. 605–623, 2019.

[16] D. Tipper, “Resilient network design: challenges and future directions,” *Telecommunication Systems*, vol. 56, no. 1, pp. 5–16, 2014.

[17] D. Santos and T. Gomes, “Controller placement and availability link upgrade problem in SDN networks,” in *RNDM*, Nicosia, Cyprus, 2019, pp. 1–8.

[18] D. Santos, T. Gomes, and D. Tipper, “Software-defined network design driven by availability requirements,” in *DRCN*, Milan, Italy, 2020, pp. 1–7.

[19] M. R. Garey, R. L. Graham, and D. S. Johnson, “Some NP-complete geometric problems,” in *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, New York, NY, USA, 1976, pp. 10–22.

[20] J.-P. Vasseur, M. Pickavet, and P. Demeester, *Network Recovery – Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. Elsevier, 2004.

[21] U. Franke, “Optimal it service availability: Shorter outages, or fewer?” *IEEE Transactions on Network and Service Management*, vol. 9, no. 1, pp. 22–33, 2012.

[22] V. Leggieri, M. Haouari, and C. Triki, “The Steiner tree problem with delays: A compact formulation and reduction procedures,” *Discrete Applied Mathematics*, vol. 164, pp. 178–190, 2014.

[23] H. Takahashi and A. Matsuyama, “An approximate solution for the Steiner problem in graphs,” *Mathematica Japonica*, vol. 24, no. 6, p. 573–577, 1980.

[24] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, “SNDlib 1.0–Survivable Network Design library,” *Networks*, vol. 55, no. 3, pp. 276–286, 2010, http://sndlib.zib.de.

[25] R. Martinez, R. Casellas, R. Vilalta, and R. M. noz, “GMPLS/PCE-controlled multi-flow optical transponders in elastic optical networks [invited],” *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 11, pp. 71–80, 2015.