Minimum Candidate Selection Algorithm for Hybrid IP/SDN Networks With Single Link Failures

Navya Vuppalapati and T.G. Venkatesh

***Abstract*—We study the problem of candidate selection in hybrid Internet Protocol (IP)/Software Defined Networks (SDN) for the case of Single Link Failures (SLFs). We propose an algorithm called Minimum Candidates Selection (MCS) that considers all possible shortest paths between all pairs of nodes in the network to minimize the number of SDN candidates needed to protect all SLFs. We demonstrate the performance of the MCS scheme in terms of repair path length and maximum link utilisation. By implementing the proposed MCS algorithm on various real world as well as random network topologies we show its advantage over the existing algorithms.**

***Index Terms*—Software defined networks, candidate selection algorithm, hybrid IP/SDN, load balancing, repair path length.**

I. Introduction

SOFTWARE Defined Networking (SDN) provides a means to sunder the control and the data planes of the traditional Internet Protocol (IP) network based devices or routers. Further, it accommodates a centralized control logic that facilitates real-time configurable network devices, and addresses network resilience, traffic management, and access control. Therefore, many organisations are upgrading their networks with SDN switches [1]. However, such a procedure incurs huge upgrading cost and is impractical to upgrade all IP routers to SDN switches at once. Thus, a feasible solution is to upgrade only a few of the IP routers with SDN switches. This kind of network consisting of both IP routers and SDN switches is referred to as a hybrid IP/SDN network [2]. The IP routers which are replaced by SDN switches are called as candidate switches. An algorithm applied to arrive at suitable candidate switches, is called as the Candidate Selection Algorithm (CSA). Much focus has been paid in the literature in proposing CSAs that can tolerate Single Link Failures (SLFs) in the network. The goal of this letter is to propose a CSA that minimises the number of candidate switches while covering all SLFs in the network.

A. Background

A survey on various approaches to the design and deployment of hybrid IP/SDN network are discussed in [3]. For a given budget restrictions, a joint problem of selecting SDN candidates and their controllers for hybrid IP/SDN networks is formulated and addressed in [4]. Incremental deployment of SDN switches in hybrid IP/SDN networks is studied in [5]. An energy-aware routing in the context of hybrid IP/SDN networks with traffic overload management is considered in [6]. A detailed survey on the evolution of the hybrid SDN has been given by Khorsandroo *et al.* [7]. An extensive survey on hybrid SDN models is given by [8]. Lin *et al.* presents systematic validation criteria and test cases for SDN application testing [9].

Henceforth we concentrate on the problem of SDN candidate set selection in a hybrid IP/SDN network aimed at covering all SLFs. The basic working principle of this class of CSAs is as follows [10]. When a link failure is found in the network, the following things happen, (i) tunneling the data from the failed link to the designated SDN switch and (ii) forwarding the data using a shortest path from the designated SDN switch to the affected destination. Both the tunneling and forwarding paths should not include the failed link.

B. Literature Survey on CSA for SLFs

Chu *et al.* [10] has proposed a CSA named as the Greedy algorithm which can be explained as follows. Initially, an SDN candidate table is formulated that has one row for each possible link failure. Row entries denote all potential candidate switches for that link failure. The candidate table has one column for each switch with entries that denote the links protected by that designated SDN candidate switch. The entries of the SDN candidate table are set as 1 or 0 based on whether the candidate switch corresponding to that row protects the link corresponding to the column. The CSA iterates as follows. First, the column sum of each node is calculated which represents the number of link failures covered by it. Then, the node with the maximum column sum is selected as SDN candidate. If there is a tie, one node having the maximum column sum is selected randomly. Next, the selected column and its associated rows are removed from the candidate table. Again one of the remaining nodes with highest column sum is picked. The process repeats until all rows are covered, i.e., all SLFs are protected. Therefore, the SDN candidate set obtained using the Greedy approach covers 100 percent link failures and is shown to achieve load balancing in the post recovery network [10].

Li *et al.* have proposed a search tree (ST) based CSA [11] that requires the same number of SDN candidates as that of the Greedy approach for most of the real network topologies [11, Table 5]. However the ST based approach has better Average Repair Path Length (ARPL)

Manuscript received 20 March 2022; revised 1 June 2022; accepted 20 June 2022. Date of publication 28 June 2022; date of current version 26 August 2022. The associate editor coordinating the review of this article and approving it for publication was R. Nejabati. *(Corresponding author: Navya Vuppalapati.)*

The authors are with the Department of Electrical Engineering, Indian Institute of Technology Madras, Chennai 600036, India (e-mail: ee17d056@smail.iitm.ac.in; tgvenky@ee.iitm.ac.in).

Digital Object Identifier 10.1109/LNET.2022.3186881

and load balancing ability as compared to the Greedy approach [11].

Finally, a Multi-Tunnel(MT) repair path based CSA has been investigated by Yang and Yeung [12] with an aim to reduce the number of the required SDN candidates. The number of candidate switches obtained using MT approach are less than those obtained with Greedy and ST algorithms as can be seen from [12, Table V] and [11, Table 5]. Also, the ARPL and load balancing ability are better for MT compared to the other two [12].

However, all the aforementioned works [10]–[12] consider only one sink tree for each node. More specifically, if there exists multiple shortest paths from a source node to a destination node, [10]–[12] choose at random only one of those shortest paths and use it in their algorithms. As a result, the number of SDN candidate switches computed using these algorithms may not be minimal. This in turn increases the implementation cost in upgrading the network. Motivated by this gap in the literature we set our aim as follows.

C. Aim and Contributions of This Letter

The primary goal of our work is to minimise the number of SDN candidates needed in a hybrid IP/SDN network to cover all SLFs. The secondary objective is to minimising either the ARPL or the Average Maximum Link Utilisation (AMLU) in the post recovery network. The contributions of this letter are listed below:

1) We propose an algorithm called Minimum Candidate Selection (MCS) algorithm to find minimum number of SDN candidates for a hybrid IP/SDN network that can tolerate all SLFs.
2) We demonstrate, using real world network topologies that the number of candidate switches obtained using our MCS approach will always be less than or equal to those obtained through other existing schemes [10]–[12].
3) MCS algorithm may yield multiple SDN candidate sets, however each candidate set can cover all SLFs.
4) Later, we study two different algorithms to select a candidate set adopting two different objectives.
5) The first algorithm, called as the MCS-ARPL selects a candidate set that minimises the ARPL. The second one called as MCS-AMLU, minimises the AMLU.

The rest of this letter is organised as follows. Section II describes the system model by taking Internet2 topology as an example. The proposed MCS algorithm is described in Section III. Section IV presents MCS-ARPL and MCS-AMLU schemes. Simulation results comparing MCS algorithm with others are presented in Section V. Finally, conclusions are drawn in Section VI.

II. System Model

Let us consider a graph denoted as $G = (V, E)$, where V is the set of vertices representing the nodes in the communication networks and E is the set of bidirectional links between the vertices. The bidirectional link between nodes i and j is denoted by (i, j). The two unidirectional links from $i \to j$ and $j \to i$ are denoted by $\langle i, j\rangle$ and $\langle j, i\rangle$ respectively. The total number of nodes and unidirectional links in the network are denoted by N and L, respectively, i.e., $N = |V|$ and $L = 2|E|$. To illustrate the proposed algorithm, we consider Internet2 topology [12] with bidirectional links as shown in Fig. 1.

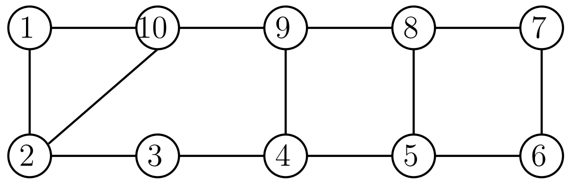

Fig. 1. Internet2 topology.

In this letter, we consider all the possible (equal length) shortest paths between any two nodes. Let $\mathcal{T}_{u,v}$ denotes the set of all possible shortest paths from node u to v. For the considered Internet2 topology, from node 1 to 6, there are four shortest paths possible which are given by:

$$T^1_{1,6} = 1 \to 2 \to 3 \to 4 \to 5 \to 6$$
$$T^2_{1,6} = 1 \to 10 \to 9 \to 4 \to 5 \to 6$$
$$T^3_{1,6} = 1 \to 10 \to 9 \to 8 \to 5 \to 6$$
$$T^4_{1,6} = 1 \to 10 \to 9 \to 8 \to 7 \to 6$$

Now we have $\mathcal{T}_{1,6} = \{T^1_{1,6}, T^2_{1,6}, T^3_{1,6}, T^4_{1,6}\}$.

III. MCS Algorithm

The proposed MCS algorithm is described in this section. The following steps are used while implementing MCS algorithm.

1) Find the affected destination set for each link failure
2) Find SDN candidate table for all link failures
3) Using the candidate table, select a candidate set with minimum number of SDN switches

1) *Affected Destinations:* We call a node d as affected destination due to the link failure $\langle i, j\rangle$, if all the shortest paths from s to d include the link $\langle i, j\rangle$, $\forall s \in V$, $s \neq d$. The affected destination set corresponding to a link failure $\langle i, j\rangle$ is denoted by either $\mathcal{AD}^{\langle i,j\rangle}$ or $\mathcal{AD}^{(l)}$, $l = 1, 2, \ldots L$.

Table I shows the affected destination sets for each link failure for both MCS and MT algorithms.[1] It can be seen from Table I that the number of affected destinations with our MCS approach are always less than or equal to that with the Greedy approach [10]. We further demonstrate this by taking the example of the failure of the link $\langle 1, 2\rangle$. The Greedy approach might have chosen $\mathcal{AD}^{\langle 1,2\rangle} = \mathcal{AD}^{(1)} = \{2, 3, 4, 5, 6\}$. Because, the only shortest path considered by the greedy approach is $T_{1,6} = 1 \to 2 \to 3 \to 4 \to 5 \to 6$. As a result, all the paths $\langle 1, 4\rangle$, $\langle 1, 5\rangle$, and $\langle 1, 6\rangle$ will have the failure link $\langle 1, 2\rangle$. With our MCS approach, we have $\mathcal{AD}^{\langle 1,2\rangle} = \{2, 3\}$. Clearly, with MCS, nodes 4, 5, and 6 are not in the affected destination set because

- For, $\langle 1, 4\rangle$: Choose $1 \to 10 \to 9 \to 4$
- For, $\langle 1, 5\rangle$: Choose $1 \to 10 \to 9 \to 4 \to 5$ or $1 \to 10 \to 9 \to 8 \to 5$
- For, $\langle 1, 6\rangle$: Choose $\to 10 \to 9 \to 4 \to 5 \to 6$, or $1 \to 10 \to 9 \to 8 \to 5 \to 6$, or $1 \to 10 \to 9 \to 8 \to 7 \to 6$.

[1]Please note that SDN candidate table is same for all Greedy, ST, and MT algorithms [10]–[12].

TABLE I
AFFECTED DESTINATIONS FOR GREEDY AND MCS, AND CANDIDATE TABLE FOR MCS

Link index l or $\langle i, j\rangle$	Affected destinations set (a_d)		SDN Candidate Table (T)									
	MCS	Greedy/MT	1	2	3	4	5	6	7	8	9	10
1, $\langle 1, 2\rangle$	2, 3	2, 3,4, 5,6	1	0	0	1	1	1	1	1	1	1
2, $\langle 1, 10\rangle$	7,8,9,10	7,8,9,10	1	1	1	1	1	1	0	0	0	0
3, $\langle 2, 1\rangle$	1	1	0	1	1	1	1	1	1	1	1	1
4, $\langle 2, 3\rangle$	3,4,5,6	3,4,5,6	0	0	0	0	0	0	1	1	1	1
5, $\langle 2, 10\rangle$	7,8,9,10	7,8,9,10	1	1	1	1	1	1	0	0	0	0
6, $\langle 3, 2\rangle$	1,2,10	1,2,10	0	0	0	1	1	1	1	1	1	0
7, $\langle 3, 4\rangle$	4,5,6,7,8,9	4,5,6,7,8,9	1	1	0	0	0	0	0	0	0	1
8, $\langle 4, 3\rangle$	2, 3	1,2,3	1	0	0	0	0	0	0	0	1	1
9, $\langle 4, 5\rangle$	5,6	5,6,7,8	1	1	0	1	0	0	1	1	1	1
10, $\langle 4, 9\rangle$	9,10	9,10	1	1	1	1	1	1	1	1	0	0
11, $\langle 5, 4\rangle$	2,3,4	1,2,3,4,9,10	1	0	0	0	1	1	1	1	1	1
12, $\langle 5, 6\rangle$	6	6,7	1	1	0	1	1	0	1	1	1	1
13, $\langle 5, 8\rangle$	8	8	1	1	1	1	1	1	1	0	1	1
14, $\langle 6, 5\rangle$	2,3,4,5	1,2,3,4, 5,8,9,10	1	0	0	0	0	1	1	1	1	1
15, $\langle 6, 7\rangle$	7	7	1	1	1	1	1	1	0	1	1	1
16, $\langle 7, 6\rangle$	6	3,4,5,6	1	1	1	1	1	0	1	1	1	1
17, $\langle 7, 8\rangle$	1,2,8,9,10	1,2,8,9,10	0	0	1	1	1	1	1	0	0	0
18, $\langle 8, 5\rangle$	5	3,4,5,6	1	1	1	1	0	1	1	1	1	1
19, $\langle 8, 7\rangle$	7	7	0	1	1	1	1	1	0	1	1	0
20, $\langle 8, 9\rangle$	1,2,9,10	1,2,9,10	0	0	1	1	1	1	1	1	0	0
21, $\langle 9, 4\rangle$	3,4	3,4,5,6	1	1	0	0	1	1	1	1	1	1
22, $\langle 9, 8\rangle$	7,8	7,8	0	1	1	1	1	1	0	0	1	0
23, $\langle 9, 10\rangle$	1,2,10	1,2,10	0	0	1	1	0	0	0	0	0	0
24, $\langle 10, 1\rangle$	1	1	0	1	1	1	1	1	1	1	1	1
25, $\langle 10, 2\rangle$	2,3	2,3	1	0	0	1	1	1	1	1	1	1
26, $\langle 10, 9\rangle$	4,5,6,7,8,9	4,5,6,7,8,9	0	1	1	0	0	0	0	0	0	0

The average number of affected destinations for all link failures is equal to 2.62 and 3.46, with MCS and Greedy, respectively. Thus, one can expect having a lower number of SDN candidates with our MCS.

2) *SDN Candidate Table:* Now, we need to select all the candidate switches that can cover a specific link failure. A node *k* is eligible to be candidate switch when the link $\langle i, j\rangle$ fails, if and only if, the following two conditions are met: (1) Among all the available shortest paths from *i* to *k*, there exists at least one path which does not include the failed link $\langle i, j\rangle$. (2) For each affected destination $d \in \mathcal{AD}^{\langle i,j\rangle}$, there exists an adjacent node *h* of *k*, such that a shortest path from *h* to *d* does not include $\langle i, j\rangle$.

Based on the above two conditions, a candidate table *T* is constructed and it is shown in Table I. The candidate table consists of *L* rows and *N* columns. Let *T*(*l*, *n*) denotes the element in l^{th} row and n^{th} column of *T*. Now, we have *T*(*l*, *n*) = 1, if the n^{th} node is a candidate switch for the l^{th} link failure, and 0 otherwise. It can be seen that, each failed link is having more number of candidate switches with our approach, as compared to the MT approach [12, Table I].

3) *Candidate Set Selection:* Let *column_sum* represents the number of failed links that are protected by a SDN candidate. A set of SDN nodes is called as candidate set if they can together cover all possible SLFs in the network. Now, we use the following approach to get all such possible candidate sets with the help of the candidate table:

- First, we check the candidate table to know if any single node is having *column_sum* = *L*, i.e., it can cover all the link failures. If yes, then each candidate set consists of only one SDN switch. We end our search after checking for all such singleton sets.

Algorithm 1 Minimum Candidate Selection Algorithm

```
Input : SDN candidate table $T$
Output: Family of SDN Candidate sets $\mathcal{C}$
1:  Initialize: $\mathcal{C} = \emptyset$
2:  for $i = 1$ to $N$ and $\mathcal{C} = \emptyset$ do
3:      Get the Combinatorial matrix $M_i$; each row is a combination
        of $i$ numbers chosen from $1, 2, , \ldots, N$. Total number of rows
        are $\binom{N}{i}$.
4:      for $j = 1$ to $\binom{N}{i}$ do
5:          Initialize: $OR\_arr$ = array of $L$ zeros
6:          for $k = 1$ to $i$ do
7:              $OR\_arr = OR\_arr \mid (M_i(j, k))^{th}$ column of $T$;
8:          end for
9:          if sum($OR\_arr$) = $L$ then
10:             $\mathcal{C} = \mathcal{C} \cup \{j^{th}$ row of $M_i\}$;
11:         end if
12:     end for
13: end for
```

- If there is no candidate switch that alone can protect all SLFs, then we check if any two nodes together can protect all the SLFs. If yes, we regard all such sets with two nodes as candidate sets. In this process,we need to check for $\binom{N}{2}$ combinations.
- If there exist no candidate set with two nodes, then we check for candidate sets with three nodes. Here, we have to check $\binom{N}{3}$ combinations.
- The same procedure is repeated until we find at least one candidate set that protects all SLFs.

Algorithm 1 shows the steps to obtain all possible candidate sets with the MCS scheme. Here $\mathcal{C}$ denotes a set that can have multiple subsets where each subset represents a candidate set. The Boolean's OR operator is denoted by |, and $\binom{N}{i}$ denotes the value of "*N* choose *i*". After following Algorithm 1 with Table I, we get 12 SDN candidate sets, given by

$$\mathcal{C} = \{\{1,3,7\},\{1,3,8\},\{1,3,9\},\{2,3,9\},\{2,4,9\},\{2,4,10\}$$
$$\{3,4,10\},\{3,5,10\},\{3,6,10\},\{3,7,10\},\{3,8,10\},\{3,9,10\}\}.$$

Please note that in Greedy and MT approaches, while selecting a SDN candidate set, one has to start by selecting a candidate with highest *column_sum*, which can lead to more number of SDN switches. The Greedy approach needs a minimum of 5 nodes and the MT needs at least 4 nodes as candidate switches for Internet2 topology. As shown above, with our approach, we just need 3 SDN candidates to cover all SLFs. Therefore, our proposed algorithm significantly reduces the implementation complexity while upgrading an IP network to Hybrid IP/SDN.

After obtaining all possible SDN candidate sets, we have to select the best among them. To pick the best candidate set we consider two metrics, namely ARPL and AMLU. In the following section, we explain how to choose a candidate set taking these two factors into account.

IV. MCS-ARPL AND MCS-AMLU ALGORITHMS

In this section, we propose two algorithms namely the MCS-ARPL and MCS-AMLU to choose the set having the minimal number of SDN candidate switches that not only covers all SLFs, but also minimises ARPL and AMLU.

1) The question we address can be stated as follows : For a candidate set $\mathcal{C}_m \in \mathcal{C}$, $1 \leq m \leq |\mathcal{C}|$, how to assign only one candidate switch for each link failure? More specifically, a link failure might be protected by more than one SDN candidates in $\mathcal{C}_m$. Thus, we have to map one element of $\mathcal{C}_m$ for a given link failure.
2) Once the first concern is resolved, we have to select the best suitable SDN candidate set.

With ARPL minimisation constraint, the first concern is addressed as below. For a given candidate set $\mathcal{C}_m$, a candidate switch $S_{m,n} \in \mathcal{C}_m$, $1 \leq n \leq |\mathcal{C}_m|$ is assigned for a link failure $\langle i, j\rangle$, if $S_{m,n}$ minimises the ARPL; here, the average is considered over the paths from i to all affected destination in $\mathcal{AD}^{\langle i,j\rangle}$, in the post recovery network. More specifically, let $RP_s^{(l)}$ be the ARPL calculated over all the affected destinations of the l^{th} link failure when node s is chosen as candidate switch. Now, $RP_s^{(l)}$ is given by

$$RP_s^{(l)} = \begin{cases} \frac{1}{|\mathcal{AD}^{(l)}|}\sum_{d\in\mathcal{AD}^{(l)}} rp_{s,d}^{(l)} & \text{if } T(l,s)=1,\\ 0 & \text{if } T(l,s)=0. \end{cases} \quad (1)$$

where $rp_{s,d}^{(l)}$ represents the RPL associated with the affected destination $d \in \mathcal{AD}^{(l)}$. Denote by $s_m^{(l)} \in \mathcal{C}_m$ the candidate switch that minimises the average repair path length when l^{th} link is failed and $\mathcal{C}_m$ is deployed. Thus, $s_m^{(l)}$ is given by

$$s_m^{(l)} = \underset{s\in\mathcal{C}_m, T(l,s)=1}{\arg\min} RP_s^{(l)}. \quad (2)$$

Therefore, each link failure is assigned with a node from a candidate set as in (2). Let us take an example with {1, 3, 7} as candidate set for the link failure $\langle 1, 2\rangle$. From Table I, it can be seen that $\langle 1, 2\rangle$ link failure can be covered by candidates 1 and 7 out of which one candidate switch has to be selected. From Table I, the affected destination set is $\mathcal{AD}^{\langle 1,2\rangle} = \{2, 3\}$. The repair paths from 1 to 2 and 3 with node 1 as candidate switch are given by $1 \to 10 \to 2$ and $1 \to 10 \to 2 \to 3$, respectively. The resulting average repair path length is 2.5 hops. Similarly, the repair paths from 1 to 2 and 3 with node 7 as candidate switch are given by $1 \to 10 \to 9 \to 8 \to 7 \to 8 \to 9 \to 10 \to 2$ and $1 \to 10 \to 9 \to 8 \to 7 \to 6 \to 5 \to 4 \to 3$, respectively; and the average repair path length is 8 hops. Thus, we select node 1 as candidate switch for $\langle 1, 2\rangle$ link failure. In a similar way, we choose one candidate from the set {1, 3, 7} for each link failure.

Now, the second concern is addressed as follows. Denote by RP_m as the ARPL calculated over all link failures, when the candidate set $\mathcal{C}_m$ is deployed. With the help of (1) and (2), RP_m is given by

$$RP_m = \frac{1}{L}\sum_{l=1}^{L} RP_{s_m^{(l)}}^{(l)} = \frac{1}{L}\sum_{l=1}^{L}\left(\underset{s\in\mathcal{C}_m, T(l,s)=1}{\arg\min} RP_s^{(l)}\right). \quad (3)$$

Finally, we select a candidate set $\mathcal{C}^*$ that minimises the total ARPL. Therefore, we have

$$\mathcal{C}^* = \underset{\mathcal{C}_m\in\mathcal{C}}{\arg\min}\ RP_m. \quad (4)$$

Table II presents all the available candidate sets and their ARPLs for Internet2 topology. Table II shows that the set

TABLE II
ARPL FOR ALL CANDIDATE SETS

S.No.	$\mathcal{C}_m$	ARPL	S.No.	$\mathcal{C}_m$	ARPL	S.No.	$\mathcal{C}_m$	ARPL
1	$\{1,3,7\}$	4.30	5	$\{2,4,9\}$	3.52	9	$\{3,6,10\}$	3.74
2	$\{1,3,8\}$	3.88	6	$\{2,4,10\}$	3.64	10	$\{3,7,10\}$	3.72
3	$\{1,3,9\}$	3.85	7	$\{3,4,10\}$	3.74	11	$\{3,8,10\}$	4.08
4	$\{2,3,9\}$	3.71	8	$\{3,5,10\}$	3.72	12	$\{3,9,10\}$	4.01

{2, 4, 9} minimises the total ARPL among all possible candidate sets. Therefore {2, 4, 9} is our final SDN candidate set.

To maintain load balancing in the recovery network, MCS-AMLU algorithm is used. MCS-AMLU algorithm is implemented exactly similar to MCS-ARPL, i.e., first we assign a suitable candidate to each link from a given candidate set; and then select the best candidate set. The only difference is that, AMLU of the repair path is minimised instead of ARPL.

V. Simulation Results

In this section, we present simulation results obtained using MATLAB to evaluate the performance of the MCS algorithm and compare it with other existing algorithms. Table III shows various real world network topologies considered for examination. This table also shows the number of nodes and bidirectional links present in each network.

Table III compares (i) the number of SDN candidates needed (ii) the ARPLs, and (iii) NAMLU for MCS, MT, ST and Greedy schemes. It is also important to show the performance of MCS scheme with real time traffic. Thus, we consider Abilene* network in Table III which is Abilene network with real traffic matrix that is given in [17]. Furthermore, random topologies (RTs) are considered in Table III which are obtained using Erdos-Renyi generator [16]. Here a 40 node RT is considered with link probabilities of p = 0.01, 0.15, and 0.20. Please note that we have averaged the results of RTs over 1000 realizations for each value of p.

It can be seen from Table III that the number of SDN candidates required by the MCS approach is always less than those with other approaches. For Abilene network, both MCS and MT outputs same number of SDN candidates. Nevertheless, for Abilene network the ARPL is smaller for MCS than MT. Further, the value of the ARPL with MCS is more for US and Germany networks than with MT. However, for both these networks, with MCS, the number of required SDN candidates are only 2; where as with MT they are 3 and 5, respectively. Thus, MCS reduces significant implementation cost while upgrading IP networks to hybrid IP/SDNs. Moreover, for RTs, it can be inferred from the table that as the p value increases the average number of links will increase and thus the number of SDN candidates reduces.

Now, to demonstrate the load balancing performance, we obtain AMLU values for the considered network topologies. Please note that, we have generated random traffic vectors whose values are uniformly distributed in [1 10]; i.e., each link in the given network is assigned a traffic value randomly from [1 10]. For each realization of a traffic vector, we calculate the AMLU over all SLFs. The experiment is repeated 1000 times. AMLU is obtained by averaging over all the experiments.

The normalized AMLU (NAMLU) values are also shown for the considered networks in Table III. Here, the

TABLE III
COMPARISON BETWEEN MCS, MT, ST, AND GREEDY ALGORITHMS

S.No.	Topology	#nodes	#links	#CS needed				ARPL				NAMLU		
				MCS	MT	ST	Greedy	MCS	MT	ST	Greedy	MCS	MT	ST
1	Internet2 [12, Fig. 1]	10	13	3	4	5	5	3.52	3.60	3.91	4.24	0.76	0.77	0.741
2	Abilene [13, Fig. 2]	11	14	3	3	5	5	3.54	3.76	3.89	4.22	0.73	0.74	0.84
3	Abilene*	11	14	3	3	5	5	3.54	3.76	3.89	4.22	0.86	0.89	0.91
4	US Network [10, Fig. 5]	14	21	2	3	3	3	3.99	3.94	4.18	4.42	0.85	0.83	0.82
5	Germany [14, Fig. 4]	17	26	2	5	5	6	5.70	3.68	3.96	4.37	1.23	0.91	0.93
6	18 node EON [15, Fig. 16]	18	44	2	4	4	4	3.30	3.37	3.45	3.87	0.81	0.82	0.86
7	Random Topology (RT)-40(0.10) [16]	40	87.9	2.4	2.6	4.1	4.4	3.9	4.04	4.23	4.50	0.95	0.96	0.80
8	RT-40(0.15)	40	119.9	2.1	2.2	3.2	3.4	3.23	3.27	3.65	4.02	0.96	0.96	0.86
9	RT-40(0.20)	40	157.1	2	2	2.1	2.2	3.01	3.01	3.23	3.68	0.9	0.9	0.9

TABLE IV
AVERAGE #CS OBTAINED WITH MCS FOR RTs ($p = 0.05$)

#Nodes	40	50	60	80	100
Avg #Links	66.75	87.50	115.60	177.85	261.8
Avg #CS	3.15	2.90	2.68	2.32	2.07

normalization is done with respect to the Greedy algorithm, so that the limitation of network bandwidth/capacity is annihilated. It can be clearly seen from the table that the MCS scheme outperforms the MT scheme, except for German and US topologies. Here a trend similar to that for ARPL can be seen, as there are only 2 candidate switches with MCS scheme.

In order to show the scalabilty of MCS scheme, we present Table IV that depicts average number of candidate switches obtained for various nodes with MCS for RTs generated with $p = 0.05$. Thus our MCS algorithm is scalable for networks with large number of nodes. Further, to study the effect of delay, we created Internet2 topology using all IP routers in the Mininet emulator followed by emulating SLFs. Random traffic was injected and the end-to-end delay was found to be 0.120 msec. In next case nodes 2, 4, and 9 of the Internet2 topology have been upgraded to SDN switches following the MCS algorithm (Table II) in the Mininet emulator. The delay was found to be 0.11 msec, thus showing that MCS offers the satisfactory level of QoS.

Finally we compare the order of complexity (OC) of different algorithms. The first common step of all the algorithms is to get the candidate table whose OC is $\mathcal{O}(LN^3)$ [10]. Let OC_{MCS}, OC_{Greedy}, OC_{MT} denote the OC for MCS,Greedy and MT respectively to get the candidate set from the candidate table. Now, by looking at $Algorithm 1$, we have $OC_{MCS} = \mathcal{O}(n_{cs}^2 \times \binom{N}{n_{cs}})$, where n_{cs} is the number of candidates in each candidate set. Further, the OC for MT and Greedy are given by $OC_{Greedy} = \mathcal{O}(n_{cs})$ and $OC_{MT} = \mathcal{O}((N-1) \times L \times n_{cs}!)$ [12], respectively. Clearly $OC_{Greedy} < OC_{MCS} < OC_{MT}$. Please note that our MCS scheme provides less number of candidate switches than MT while having lower OC than the later.

VI. CONCLUSION

In this letter, we have proposed an algorithm called MCS for the problem of candidate set selection for hybrid IP/SDN networks with SLFs. MCS algorithm reduces the number of SDN candidates needed as compared to other existing schemes for all network topologies and has lower ARPL and AMLU. In future we can extend our work by considering energy efficiency and delay constraints. Further, this letter can be extended by considering multi-link failures.

REFERENCES

[1] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, 2013.

[2] S. Vissicchio, L. Vanbever, L. Cittadini, G. G. Xie, and O. Bonaventure, "Safe update of hybrid SDN networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1649–1662, Jun. 2017.

[3] X. Huang, S. Cheng, K. Cao, P. Cong, T. Wei, and S. Hu, "A survey of deployment solutions and optimization strategies for hybrid SDN networks," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1483–1507, 2nd Quart., 2019.

[4] Z. Guo, W. Chen, Y.-F. Liu, Y. Xu, and Z.-L. Zhang, "Joint switch upgrade and controller deployment in hybrid software-defined networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1012–1028, May 2019.

[5] M. Huang and W. Liang, "Incremental SDN-enabled switch deployment for hybrid software-defined networks," in *Proc. Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2017, pp. 1–6.

[6] N. Huin, M. Rifai, F. Giroire, D. L. Pacheco, G. Urvoy-Keller, and J. Moulierac, "Bringing energy aware routing closer to reality with SDN hybrid networks," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 4, pp. 1128–1139, Dec. 2018.

[7] S. Khorsandroo, A. G. Sánchez, A. S. Tosun, J. Arco, and R. Doriguzzi-Corin, "Hybrid SDN evolution: A comprehensive survey of the state-of-the-art," *Comput. Netw.*, vol. 192, Jun. 2021, Art. no. 107981.

[8] Sandhya, Y. Sinha, and K. Haribabu, "A survey: Hybrid SDN," *J. Netw. Comput. Appl.*, vol. 100, pp. 35–55, Dec. 2017.

[9] Y.-D. Lin, Y.-K. Lai, Y.-L. Tsou, Y.-C. Lai, E.-C. Liou, and Y. Chiang, "Generic validation criteria and methodologies for SDN applications," *IEEE Syst. J.*, vol. 13, no. 4, pp. 3909–3920, Dec. 2019.

[10] C.-Y. Chu, K. Xi, M. Luo, and H. J. Chao, "Congestion-aware single link failure recovery in hybrid SDN networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2015, pp. 1086–1094.

[11] N. Li, Y. Shi, Z. Zhang, J. F. Martinez, and X. Yuan, "Search-tree based SDN candidate selection in hybrid IP/SDN network," in *Proc. IEEE Int. Conf. Netw. Protocols (ICNP)*, 2020, pp. 1–6.

[12] Z. Yang and K. L. Yeung, "SDN candidate selection in hybrid IP/SDN networks for single link failure protection," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 312–321, Feb. 2020.

[13] M. M. Amble, P. Parag, S. Shakkottai, and L. Ying, "Content-aware caching and traffic management in content distribution networks," in *Proc. IEEE INFOCOM*, 2011, pp. 2858–2866.

[14] S. Turk, S. Sulaiman, A. Haidine, R. Lehnert, and T. Michaelis, "Approaches for the migration of optical backbone networks towards carrier Ethernet," in *Proc. IEEE Globecom Workshops*, 2009, pp. 1–6.

[15] F. Lezama, G. Castñón, and A. M. Sarmiento, "Routing and wavelength assignment in all optical networks using differential evolution optimization," *Photon. Netw. Commun.*, vol. 26, no. 2, pp. 103–119, 2013.

[16] P. Erdös and A. Rényi, "On random graphs," *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[17] P. Tune and M. Roughan, "Internet traffic matrices: A primer," in *Proc. Recent Adv. Netw.*, vol. 1, 2013, pp. 1–56.