Single-Link Failure Recovery With or Without Software-Defined Networking Switches

Dawei Li*, Jie Wu†, and Dajin Wang*
*Department of Computer Science, Montclair State University, Montclair, NJ, 07043
†Center for Networked Computing, Temple University, Philadelphia, PA, 19121
Email: dawei.li@montclair.edu, jiewu@temple.edu, wangd@mail.montclair.edu

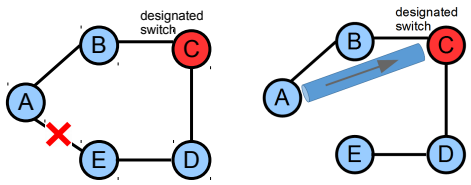
***Abstract*—In this paper, we consider IP fast recovery from single-link failures in a given network topology. The basic idea is to replace some existing routers with a designated switch. When a link fails, the affected router will send all the affected traffic to the designated switch (through pre-configured IP tunnels), which will deliver the affected traffic to its destination without using the failed link. The goal of the approach is to achieve faster failure recovery than traditional routing protocols that employ reactive computing upon link failures. Software-Defined Networking (SDN) switches can serve as the designated switches because they can flexibly redirect affected traffic to other routes, instead of only to the shortest paths in the network. However, SDN switches are very expensive. Our objective is to minimize the number of SDN switches needed and to guarantee that the network can still recover from any single-link failure. For networks with uniform link costs, we show that using normal non-SDN switches with IP tunneling capability as designated switches can guarantee recovery from any single-link failure. For networks with general link costs, we show that by using SDN switches only when necessary, we can reduce the total number of SDN switches needed compared to an existing work.**

***Index Terms*-Software-defined networking (SDN), failure recovery, shortest paths, equal cost multi-path (ECMP), IP tunneling.**

I. Introduction

Traditional IP networks are complex and hard to manage. Software-Defined Networking (SDN) [1], which separates the network's control logic from the underlying routers and switches, has become an important technology that enables flexible traffic control and network management. SDN switches are usually very expensive; thus, the deployment of SDN is usually a gradual process. In practice, we see networks where traditional IP routers and SDN switches coexist. This kind of networks are referred to as hybrid SDN networks [2], [3]. SDN switches also have IP router functionalities. In the rest of the paper, we use routers and switches interchangeably.

In this paper, we consider using SDN switches to improve the network's fault tolerance. Specifically, given an IP network topology, we consider replacing some of the IP routers with designated switches, which may or may not be SDN switches, in order to allow the network to recover from any single-link failure. Because SDN switches are expensive, we aim to minimize the number of SDN switches needed, in order to minimize the cost of providing such fault tolerance. We want to emphasize that the fault tolerance we want to achieve here is very different from that of traditional routing protocols. When a failure happens, in traditional routing protocols, the failure information needs to be propagated to a considerable part of the network, and reactive computing is usually required before the network stablizes. In our design, the affected traffic can be immediately redirected to designated switches; thus, the network can recover from the failure very fast.



(a) Link $e_{A,E}$ fails. (b) IP tunneling.

Fig. 1. Motivational example. We set up a pre-configured IP tunnel from A to C. When link $e_{A,E}$ fails, traffic from A cannot reach E or D through shortest paths. With IP tunneling, affected traffic at A will be first redirected to C through the IP tunnel from A to C. After that C can further deliver the traffic to its destinations.

An existing work considers using SDN switches to replace some legacy IP routers, so that the network can recover from any single-link failure [2]. When a link failure occurs, the router with the failed link will redirect affected traffic to an SDN switch, which can efficiently figure out a next hop switch, from which the shortest path to the destination can avoid the failed link. The SDN switch used for failure recovery is called a *designated* switch; the objective is to minimize the number of SDN switches in the network, while still allowing the network to recover from any single-link failure.

We find that the designated switch does not always need to be an SDN switch. In some cases, we can use a traditional non-SDN designated switch to replace a legacy IP router and then the network can recover from single-link failures. We can succeed in doing so if 1.) the shortest path from the affected router to the designated switch does not include the failed link, and 2.) the shortest path from the designated switch to the affected destination does not include the failed link. In these cases, the designated switch only needs to have IP tunneling capability but does not need to be an SDN switch. IP tunneling capability is common in non-SDN switches [4]; this makes it possible to use non-SDN switches as designated switches.

A. Motivational Example

We use an example, shown in Fig. 1, to illustrate our idea. Assume that all links in the network have the same cost. Consider the case when the link between A and E fails. If A is the source router, traffic from A to E and traffic from A to D will be affected by this link failure.

The authors in [2] propose using SDN switches to replace some routers to improve the network's fault-tolerance. In this example, an SDN switch is used to replace the router either at B or C. If the SDN switch replaces the router at C, A will

forward affected traffic to the SDN switch at C. The SDN switch at C can choose its neighbor D as the next hot switch, and from there, the traffic can follow the shortest path to E and D without using the failed link $e_{A,E}$.

Alternatively, we can use a traditional non-SDN switch with IP tunneling capability to replace the router at C, to help the network recover from the failure. If we have a designated switch at C, we can setup an IP tunnel from A to C going through B. When the link $e_{A,E}$ fails, we can let the traffic at A go to C using the IP tunnel $A \rightarrow B \rightarrow C$. Then from C, the traffic can still take shortest paths, $C \rightarrow D$ and $C \rightarrow D \rightarrow E$ to arrive at destinations D and E, respectively. Since IP tunneling capability is quite common in modern routers, we actually do not need to physically replace the router; we only need to change the configuration of the router.

From this simple example, we can draw an important intuition: if the network can recover from single-link failures using only non-SDN designated switches with IP tunneling capability, we can further reduce the total number of SDN switches needed to guarantee that the entire network can recover from any single-link failure.

Our contributions in this paper can be outlined as follows:

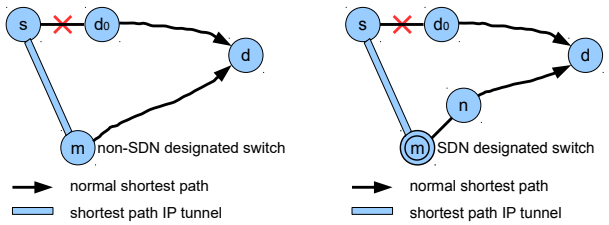
- First, we find that SDN switches are not always needed to recover from a link failure. Sometimes, a traditional non-SDN switch with IP tunneling capability is sufficient to recover the network from a single-link failure.
- Second, for networks with uniform link costs, we show that by using only non-SDN switches as designated switches, it is guaranteed that the entire network can recover from any single-link failure.
- Third, for networks with general link costs, we prove that by using SDN switches as designated switches, it is guaranteed that the entire network can recover from any single-link failure. Besides, by using SDN switches as designated switches only when necessary, we can further reduce the number of SDN switches needed compared to an existing work [2].

The rest of the paper is organized as follows. The problem description, our design approach, and some preliminary results are presented in Section II. In Section III, we consider minimizing the number of SDN switches needed while guaranteeing recovery from any single-link failure. Supporting simulations are conducted in Section IV. Conclusions are made in Section V.

II. Problem, Design, and Preliminaries

A. Problem Description

Given a network topology, our goal is to replace a subset of routers with designated switches to recover the network from any single-link failure. The fault tolerance we want to achieve here is very different from traditional routing protocols. In traditional routing protocols, such as RIP and OSPF, when a failure happens in the network, it takes a long time for the failure information to be propagated into the network, and some lengthy reactive computation may be required for the network to stablize. In our design with designated switches, affected traffic can be immediately redirected to the designated switch upon detection of the link failure.



(a) Use non-SDN designated switch. (b) Use SDN designated switch.

Fig. 2. An example for recovering from a single-link failure. In figure (a), m is a non-SDN designated switch. In figure (b), m is an SDN designated switch. n is a direct neighbor of m.

In our consideration, a practical design should have the following features. First, it should take advantage of protocols available on today's routers; it means that routers without SDN functionality should always forward the traffic through shortest paths. When multiple shortest paths are available, Equal-Cost Multi-Path (ECMP) routing can be utilized to achieve traffic balance in the network [5]. Also, ECMP itself can avoid immediate failed links when other shortest paths are available. Second, the network should recover from any single-link failure immediately when the failure is detected.

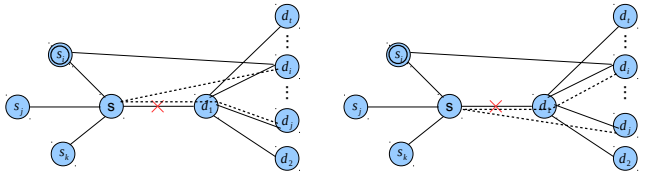
Our network under consideration can be modeled as an undirected simple bi-connected graph. A bi-connected graph is a graph which will remain connected if any one vertex were to be removed [6]. The bi-connected graph model for a network guarantees that the network will still be connected upon any single-link failure. This is the foundation for our single-link failure recovery scheme.

B. Overall Design

In our approach, each interface of a router is configured to have a backup IP tunnel to provide failover upon detecting a link failure on its interface. The IP tunnel is established between the router and a designated switch, which may or may not be an SDN switch. Whenever a link failure is detected, the router, which is directly connected to the failed link, would immediately encapsulate and forward all affected traffic to its designated switch through the backup IP tunnel. As shown in Fig. 2(a) and Fig. 2(b), when link e_{s,d_0} fails, router s will redirect all affected traffic to a designated switch at m. Because from s to m, all traversed routers in the IP tunnel are normal routers, the IP tunnel should be a shortest path. The designated switch m takes further actions to make sure that the traffic gets to its destination without using the failed link.

We use two types of designated switches to recover the network from single-link failures: *non-SDN designated switch*, and *SDN designated switch*. Non-SDN designated switches can only set up IP tunnels. From the affected router to the non-SDN designated switch, the IP tunnel must be a shortest path; from the non-SDN designated switch to the destination switch, the path must also be a shortest path.

SDN designated switches have more flexibility in terms of redirecting affected traffic. From the affected router to the SDN designated switch, the IP tunnel must still be a shortest path; from the SDN designated switch to the destination switch,



(a) Case I (b) Case II

Fig. 3. Two possible cases of the shortest paths for networks with general link costs. We use an SDN designated switch at s_i for s when e_{s,d_1} fails.

the path is not necessarily a shortest path. The SDN switch can select any of its neighbor switches, and then from the selected neighbor switch to the destination, the traffic must take a shortest path.

We require that the assignment of designated switches to be destination independent. The advantage of this requirement is that the complexity of configuring failover is minimized since routers do not need to deal with each individual destination. Therefore, for a designated switch to be eligible, it must be able to accommodate all the possible destinations tunneled from an affected router.

C. Preliminaries

In this section, we investigate important characteristics of the problem. For networks with uniform link costs, there exist several methods for recovering any single link failure by just using IP routers' tunneling capability. Examples include IP Fast Reroute (IPFRR) Not-Via [7] and a more recent method proposed in [8]. In other words, we do not need any SDN switch to recover from any single link failure.

Next, we consider networks with general link costs. Given the failed link $e_{s,d}$, the nodes in the graph can be partitioned into two parts. The first part consists of all the nodes to which the shortest paths from s include the failed link $e_{s,d}$. The second consists of all the nodes to which the shortest paths from s do not include the failed link $e_{s,d}$. Denote all nodes in the first part by $\mathcal{D} = \{d_1, d_2, \cdots, d_k\}$[1]. Denote all nodes in the second part by $s_1, s_2, \cdots, s_l$, where $s_1 = s$. It is easy to find examples where networks with general link cost cannot recover from all single-link failures with only non-SDN designated switches. Next, we show that using SDN designated switches, it is guaranteed that networks with general link costs can recover from any single-link failure.

First, there exists at least one node in the first part that has a link to a node in the second part. Otherwise, it violates the properties of a bi-connected graph. Denote the node in the first part as d_i and the node in the second part as s_i. We can then choose s_i as the designated SDN switch for s_1 when $e_{s,d}$ fails, and choose d_i as the next hop switch for all the destination nodes in the first part.

Lemma 1: The shortest path from d_i to d_j will never use the failed link $e_{s,d}$.

Proof: We prove this by contradiction. Assume that the shortest path from d_i to d_j includes $e_{s,d}$ as one link in it. We have two cases to investigate.

[1]It is possible for $\mathcal{D}$ to be empty, for example, when the cost of $e_{s,d}$ is very high, and no shortest path includes it. In this case, the network can be considered unaffected. In the rest of the paper, we will not consider this special case.

In the first case, as shown in Fig. 3(a), the shortest path p_{d_i,d_j} consists of the shortest path from d_i to s, $p_{d_i,s}$, the link, $e_{s,d}$, and the shortest path from d to d_j, p_{d,d_j}. However, since d_i is affected by the failed link $e_{d,s}$, there is a shortest path from s to d_i that consists of $e_{s,d}$. The length of the shortest path $p_{d_i,s}$ is equal to that of $e_{s,d}$ + p_{d,d_i}. Also, the length of the path $e_{s,d}$ + p_{d,d_j} is greater than that of p_{d,d_j}. Then, the length of p_{d_i,d_j} is greater than the length of $p_{d_i,d}$ + p_{d,d_j}, which violates the fact that p_{d_i,d_j} is a shortest path.

In the second case, as shown in Fig. 3(b), the shortest path p_{d_i,d_j} consists of the shortest path from d_i to d, $e_{d,s}$, and the shortest path from s to d_j. However, since d_j is affected by the failed link $e_{d,s}$, there is a shortest path from s to d_j that consists of $e_{s,d}$. The length of the shortest path p_{s,d_j} is equal to that of $e_{s,d}$ + p_{d,d_j}. The length of $p_{d_i,s}$ is greater than that of $p_{d_i,d}$. Then, the length of p_{d_i,d_j} is greater than the length of $p_{d_i,d}$ + p_{d,d_j}, which again violates the fact that p_{d_i,d_j} is a shortest path.

Thus, the shortest path from d_i to d_j will never use the failed link $e_{s,d}$. ■

Theorem 1: A bi-connected network can recover from any single-link failure by using SDN designated switches.

Proof: The theorem follows from Lemma 1, since the analyses in Lemma 1 applies to any general link in the network. ■

III. Minimizing the number of SDN switches

In this section, we consider minimizing the number of SDN switches needed in detail. Recall that networks with uniform link costs can recover from any single-link failure by using only non-SDN designated switches. From now on, we focus on networks with general link costs.

Definition: A link is called a Non-SDN Recoverable Link (NSRL) if the failure of this link does not need an SDN switch to be the designated switch.

Notice that when a link fails, the two end nodes of the link can be configured to take different actions. Thus, for each link failure, we need to consider two directions of the failure.

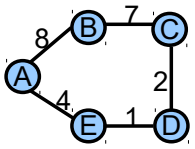
A. Collecting Non-SDN Recoverable Links

The first step of our approach is to identify all directed NSRLs. Given a link failure, say $\vec{e}_{s,d}$, we assume that the source is s, and we denote all the destinations that are affected by this link failure as $\mathcal{D} = \{d_i\}$. The link, $\vec{e}_{s,d}$, can be regarded as an NSRL if and only if $\forall d_i \in \mathcal{D}$ there exists a switch in the network, m ($m \notin \mathcal{D}$), such that the shortest path from s to m, $p_{s,m}$ and the shortest path from m to d_i, p_{m,d_i} both do not include the failed link.

We can first collect all such NSRLs, denoted as L_{NSRL}. We denote all link failures in the network as L_U, where $L_U = \{\vec{e}_{s,d} | e_{s,d} \in E\} \cup \{\vec{e}_{d,s} | e_{s,d} \in E\}$. After collecting NSRLs, we need to consider recovering from failures in $L_R = L_U - L_{NSRL}$ using SDN switches.

Fig. 4 shows a simple example on how to collect the NSRLs in a network. The link costs of $e_{A,B}$, $e_{B,C}$, $e_{C,D}$, $e_{D,E}$, and $e_{E,A}$ are 8, 7, 2, 1, and 4, respectively. There are five links in total. Considering the link directions, there are 10 link failure cases. The table in Fig. 4 shows the information for

Link Failure	Source	Affected Destinations	NSRL
A->B	A	B	Y
B->A	B	A	Y
B->C	B	C, D, E	Y
C->B	C	B	Y
C->D	C	D, E, A	N
D->C	D	C, B	Y
D->E	D	E, A	N
E->D	E	D, C, B	N
E->A	E	A	Y
A->E	A	E, D, C	Y



Link Failure	Source	Affected Destinations	SDN candidate locations				
			A	B	C	D	E
C->D	C	D, E, A	0	1	0	0	0
D->E	D	E, A	0	1	1	0	0
E->D	E	D, C, B	1	0	0	0	0

Fig. 5. Illustration of candidate table construction.

each of the link failures. Each link failure is represented by a row in the table. The first column is the failed link; the second column is the source node; the third column is the destinations that are affected by this link failure; the fourth column indicates whether the network can recover from this link failure only using non-SDN designated switches. A "Y" in the fourth column means that the failed link is an NSRL; an "N" means that it is not.

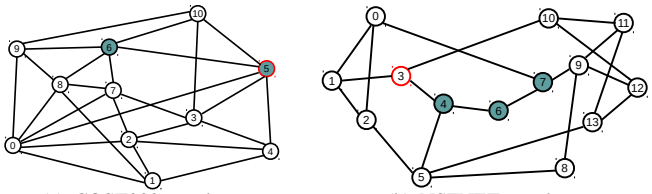
Consider the link failure of $\vec{e}_{B,C}$. Traffic from B to C, D, and E will be affected. This is because the shortest paths from B to C, from B to D, and from B to E all include $\vec{e}_{B,C}$ as a link. Though three destinations are affected, the network can still recover from this link failure by using a non-SDN switch at node A. So, the traffic from B destined at C, D, and E will be redirected to node A. Since the shortest paths from A to C, from A to D, and from A to E do not include $\vec{e}_{B,C}$ as a link. The traffic can successfully arrive at the destinations using shortest paths. Thus, link $\vec{e}_{B,C}$ is an NSRL.

B. Using SDN Switches for Remaining Link Failures

After identifying all NSRLs, we need to consider using SDN switches to recover from failures in $L_R = L_U - L_{NSRL}$. The problem of minimizing the number of SDN switches can be formulated as a binary integer programming problem which is a well-known NP-complete problem [2] [9]. We use a heuristic algorithm for this problem with polynomial time complexity. It includes two phases: candidate table construction and column selection.

For candidate table construction, the goal is to identify eligible candidate SDN locations for each link failure in L_R. Fig. 5 shows the candidate table for the topology in Fig. 4. Note that, we only consider links in L_R because all links in L_{NRSL} do not require SDN switches for failure recovery. For each link failure $i \rightarrow j$, all affected destinations are again denoted by $\mathcal{D}$. A node $k \notin \mathcal{D}$ is an eligible candidate location for an SDN designated switch for this failure if it meets the following conditions: 1) the shortest path from i to k does not include the failed link, and 2) for each affected destination $d \in \mathcal{D}$, k must have at least one neighbor m, such that the shortest path from m to d does not include the failed link.

If node k meets the above conditions, it is labeled "1," otherwise, it is labeled "0." After constructing the candidate table, the problem is reduced to a column selection problem. The goal is to select the least number of SDN locations (columns) so that all failures in L_R (all rows in the table) are covered. For the selected columns to cover all rows, the sum of each row should be greater than 0.



(a) COST239 topology (b) NSFNET topology

Fig. 6. Specific network topologies.

A greedy approach is used to select the columns. For each column, we add up all the elements and the sum is called the weight of the column. This weight indicates the number of failures that can be covered if an SDN switch is placed at the corresponding location. We then choose the column with the largest weight and remove this column as well as rows that are covered by this column. The weights are then updated for the remaining columns and rows, and another column with the largest weight is selected. We repeat this process until all the rows are removed which means that all the possible link failures are covered. These selected columns corresponds to the locations of routers that we should replace with SDN switches.

IV. Evaluations

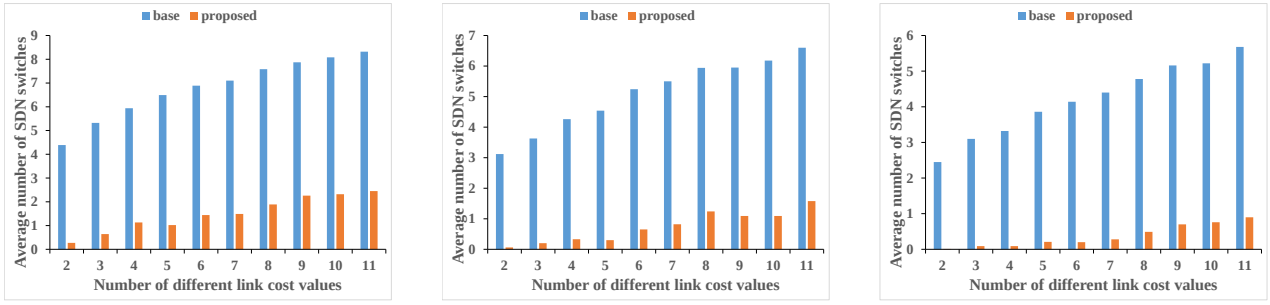
We conduct simulations with different topologies to compare the number of SDN switches needed using different methods. We call the existing method in [2] the "*base*" method, where all designated switches are SDN switches. We call ours the "*proposed*" method, where we use SDN switches as designated switches only when required. Notice that, for networks with uniform link costs, our *proposed* method requires no SDN switches, in this section, we only conduct comparisons for networks with general link costs.

A. Basic Network Topologies: Specific Examples

Two practical network topologies are used for the evaluation: the 11-node COST239 topology [10] with 26 links and the 14-node NSFNET topology [11] with 21 links.

We assign different costs to the links in the network. We start from a network with uniform link costs, i.e., all links with a cost value of 1. Then, we pick some links and assign different cost values to them. In our simulations, we find that, if we select a small number of links to assign different link costs, it is probable that our *proposed* method still does not require any SDN switches to achieve the desired fault-tolerance. In the following, we will present several examples for each of the three topologies, where our *proposed* method does require some SDN switch(es) to achieve the desired fault-tolerance.

For the COST239 topology, if the costs of $e_{0,1}$, $e_{1,4}$, $e_{3,4}$, $e_{3,10}$ and $e_{6,9}$ are 2, the costs of $e_{2,4}$, $e_{3,5}$, $e_{5,6}$, $e_{6,7}$ and $e_{7,8}$ are 3, the costs of $e_{6,10}$ is 4, while all other links' costs remain 1, then, the *base* method will replace the switches at 6 and 5 with SDN switches as shown in Fig. 6(a), and our *proposed* method only need to replace the switch at 5 with one SDN switch. For the NSFNET topology, if the costs of links $e_{0,1}$, $e_{3,10}$, $e_{4,5}$, $e_{5,8}$, $e_{6,7}$ and $e_{6,7}$ are 2, the cost of link $e_{8,9}$ is 3, the cost of link $e_{4,6}$ is 4, while all other links' costs remain 1,



(a) Average node degree = 4.512 (b) Average node degree = 6.155 (c) Average node degree = 7.949

Fig. 7. Simulations for networks with 100 nodes.

then, the *base* method will replace the switches at 4, 7, and 6 with SDN switches, as shown in Fig. 6(b), and our *proposed* method only needs to replace the switch at 3 with one SDN switch.

B. Randomly Generated Network

We also conduct simulations for various randomly generated networks. We aim to investigate how the node degree of a network influences the performances of both methods. We design our simulation groups as follows. First, we fixed the number of nodes in the network. They, we randomly generate 100 networks with a certain average node degree. For all these 100 networks, we further conduct 10 groups of simulations. In the first simulation group, links in the 100 networks choose random link cost values from $\{1, 2\}$. In the 2nd simulation group, network link costs choose random values from $\{1, 2, 3\}$, and so on. In the 10th group, network link costs choose randomly value from $\{1, 2, \cdots, 11\}$.

For a network with a fixed number of nodes, we intend to generate links such that the network will have some fixed node degree, i.e., 4, 6, or 8, by giving a fixed parameter. However, the links are randomly generated, and the resulting number of links is not fixed. Thus, each network's average node degree is not exactly equal to the given parameter.We conduct simulations for networks with 100 nodes.

The simulation results are presented in Fig. 7. In Fig. 7(a), Fig. 7(b), Fig. 7(c), the average node degrees for the networks are 4.512, 6.155, and 7.949, respectively. We can see that, besides the network link costs, the network node degree has a significant influence on the number of SDN switches needed using both methods. In Fig. 7(a), the average number of SDN switches can be greater than 4 using the *base* method; in Fig. 7(c), this number drops to about 2. Similarly, in Fig. 7(a), the average number of SDN switches can be greater than 2 using our *proposed* method; in Fig. 7(c), this number drops to less than 1, meaning that when the network node degree is 8, many of the networks do not need an SDN designated switch to achieve single-link failure recoverability. From these simulation results, we can still see the influence of network link cost distribution on the number of SDN switches needed. When the deviation of the link cost values is small, i.e., when the link costs choose values from a small set of values, most of the networks do not require SDN switches to have the desired fault-tolerance using our *proposed* method, as shown in the first three simulation groups of Fig. 7(c). In all simulation cases, our proposed method reduces the number of SDN switches needed significantly.

V. Conclusions

In this paper, we consider IP fast recovery from single-link failures. By redirecting traffic from the failed link to designated switches through pre-configured IP tunnels, our proposed approach is able to react to the failures very fast. For networks with uniform link costs, we show that we can use normal non-SDN switches with IP tunneling capability as designated switches to recover the network from any single-link failure. For networks with general link costs, we show that using SDN switches as designated switches can always allow the network to recover from any single-link failure. By using SDN switches only when necessary, we can further reduce the total number of SDN switches needed compared to an existing method. Extensive simulations have been conducted to verify the applicability of our approaches.

References

[1] D. Kreutz, F. M. V. Ramos, P. E. Verssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, pp. 14–76, Jan. 2015.

[2] C. Y. Chu, K. Xi, M. Luo, and H. J. Chao, "Congestion-aware single link failure recovery in hybrid sdn networks," in *IEEE Conference on Computer Communications (INFOCOM)*, pp. 1086–1094, Apr. 2015.

[3] H. Xu, X. Y. Li, L. Huang, H. Deng, H. Huang, and H. Wang, "Incremental deployment and throughput maximization routing for a hybrid sdn," *IEEE/ACM Transactions on Networking*, vol. PP, no. 99, pp. 1–15, 2017.

[4] S. Kini, S. Ramasubramanian, A. Kvalbein, and A. F. Hansen, "Fast recovery from dual-link or single-node failures in ip networks using tunneling," *IEEE/ACM Transactions on Networking*, vol. 18, pp. 1988–1999, Dec. 2010.

[5] A. Iselt, A. Kirstadter, A. Pardigon, and T. Schwabe, "Resilient routing using mpls and ecmp," in *Workshop on High Performance Switching and Routing (HPSR)*, pp. 345–349, 2004.

[6] P. E. Black, "biconnected graph," *Dictionary of Algorithms and Data Structures, Paul E. Black, ed., U.S. National Institute of Standards and Technology*, Dec. 2004. https://xlinux.nist.gov/dads/HTML/biconnectedGraph.html.

[7] A. Atlas and A. Zinin, "IP fast reroute using not-via addresses," in *IETF Draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-05*, 2010.

[8] S. Bryant, C. Filsfils, S. Previdi, M. Shand, and N. So, "Remote loop-free alternate (LFA) fast reroute (FRR)," in *RFC 7490*, Apr. 2015.

[9] R. M. Karp, "Reducibility among combinatorial problems," in *Springer*, 1972.

[10] P. Batchelor, *Ultra High Capacity Optical Transmission Networks: Final Report of Action COST 239*. Faculty of Electrical Engineering and Computing, 1999.

[11] D. Banerjee and B. Mukherjee, "Wavelength-routed optical networks: linear formulation, resource budgeting tradeoffs, and a reconfiguration study," *IEEE/ACM Transactions on Networking*, vol. 8, pp. 598–607, Oct. 2000.