ENHANCING NETWORK SURVIVABILITY USING ROUTING AND LINK WEIGHT ASSIGNMENT TO AVOID CONGESTION

P.-K. TSENG, W.-H. CHUNG, C.-F. WU & C.-H. WU
Research Center for Information Technology Innovation Academia Sinica, Taiwan.

ABSTRACT

The modern wired networks transport high data rates, where a single-network failure often causes enormous packet losses. To provide adequate protection for all source–destination pairs remains a critical issue in modern networks. In this article, a high survivability Internet Protocol (IP) fast-reroute scheme against single-link and single-node failures is first described, and then a Mixed Integer Non-linear Programming (MINP) is formulated to determine link weights so as to maximize the survivability of networks running the IP fast-reroute scheme. A simulated annealing-based routing and weight assignment scheme is proposed to approximate the optimal solution of the MINP. The simulations demonstrate that the proposed scheme can improve network survivability rate up to 83–100% for single-link failures, and 78–100% for single-node failures, and achieve reasonably good load balancing in five benchmark networks.

Keywords: IP fast-reroute, load balance, loop-free, single-link and single-node failures, survivability.

1 INTRODUCTION

Network failures often cause service interruptions and tremendous packet losses in the modern high data-rate network [1,2]. A total recovery process for current link state routing protocols, such as the open shortest path first (OSPF) [3], may take up to tens of seconds to re-converge [4]. To reduce the amount of packet loss and service interruption during this period, certain work must be done to speed up the convergence of link state routing protocols [5–8]. Many quality-of-service (QoS)-sensitive and time-critical applications demand even faster convergence and higher survivability in modern networks.

In this article, an IP fast-reroute scheme, termed as the *Unaffected Alternate Selection* (UAS) scheme, is proposed against single-link and single-node failures within a network running OSPF link state routing protocol. According to UAS, each router differentiates the routing capacities of adjacent neighbors, pre-selects one of them to be an alternative backup next hop for handling single-link or single-node failure, and builds a backup routing table, including selected backup next hops in advance. The UAS scheme achieves high network survivability, avoids traffic congestion, and guarantees routing loop-free property during the entire healing process.

Once a router detects a failure, it diverts the affected traffic to the alternative backup next hop indicated by the pre-computed backup routing table immediately without disturbing the regular flows. This local reaction process guarantees fast switching and reduces failure recovery time. The affected packets are marked with a 1-bit 'tag', so that the routers on the backup path know to use backup next hop to forward these affected packets. When the duration of failure exceeds a specific time interval, a normal reconvergence procedure (such as OSPF reconvergence) is triggered. The proposed UAS scheme continues packet forwarding during the reconvergence procedure. The possible micro-loops during reconvergence procedure can be handled easily as described in refs [9,10].

The underlying survivable IP routing and link weight assignment problem is formulated as a Mixed Integer Non-linear Programming (MINP) problem to maximize network survivability

ISSN: 2041-9031 (paper format), ISSN: 2041-904X (online), http://www.witpress.com/journals
DOI: 10.2495/SAFE-V5-N1-29-47

achievable by UAS or well-known IP fast-rerouting approaches, Loop-Free Alternate (LFA), and U-Turn. There are several optimization solvers for solving this class of problems near optimally in an acceptable amount of time for many instances of the problem. A *Simulated Annealing based Routing and Weight Assignment* (SARWA) scheme is also proposed to obtain an acceptably good solution of MINP in shorter time. The link weight determined by SARWA not only supports working path routing but also maximizes network survivability under single-link and single-node failures.

The major contribution of this article includes the proposed IP fast reroute scheme and link weight assignment. It ensures loop-free route during the healing process. Network congestion is also considered by the proposed scheme. The idea of having the IP fast reroute scheme with pre-designed backup routing table is also new.

The remainder of this article is organized as follows. Related work is summarized in Section 2. The proposed UAS is described in Section 3. Problem formulations are described in Section 4. The proposed SARWA is introduced in Section 5. In Section 6, the experimental results are presented and the performance comparisons with other well-known schemes are shown. Finally, concluding remarks are given in Section 7.

2 RELATED WORK

Many IP fast-rerouting schemes have been proposed in the literature, e.g. Equal-Cost Multi-Path (ECMP) [11], LFA [12], and U-Turn [13]. IP fast rerouting schemes focus on finding an alternative route (rather than finding an optimal route) as soon as possible while avoiding any routing loops.

ECMP [11] is a routing strategy that aims to offer increased bandwidth by load-balancing traffic over multiple paths. ECMP has been applied to provide a solution for fast rerouting in which multiple paths serve as backup paths for each other. However, the weight on each link has to be designed on a case-by-case basis, making ECMP infeasible for practical deployment on a large-scale network.

The LFA scheme [12] is performed on adjacent routers close to the failed link to avoid frequent route-flipping during the healing process. The idea of LFA is to find a loop-free shorter distance alternative router and redirect traffic to that router when a link failure occurs. However, the existence of a loop-free alternative route is topology dependent and the overall survivability may be low. U-Turn [13] aims to improve the low survivability in LFA by allowing packets to travel back to the same link twice. The scheme has two variants: Implicit and Explicit U-Turns. The Implicit U-Turn scheme requires no modification to the packets but normally requires interface-specific reverse path forwarding verification [14]. Conversely, Explicit U-Turn marks packets explicitly and thus requires a special marking mechanism. Because of the cost of greater complexity and implementation effort, U-Turn is not able to provide adequate fault tolerance capability for network failures [14,15].

Tunnel-based schemes presented in refs [16–18] also aim to achieve high network survivability. According to these schemes, when a router detects an adjacent link failure, the router selects an intermediate router, encapsulates the packets carried on the failed link, and reroutes them to the intermediate router to bypass the failure. Once the intermediate router receives these encapsulated packets, the router decapsulates the packets and forwards them according to corresponding destinations via normal routes. The intermediate router must be selected carefully to avoid routing loops. Tunnels [16], Not-via address [17], and Protection Graphs Construction [18] adopt this design concept. Encapsulation and decapsulation of packets introduce extra burden on routers in the network.

According to *Backup Routing Table* (BRT)-based recovery schemes introduced in refs [19–27], each router pre-computes backup routing tables before any failure occurs. Each backup table is assigned a dedicated identity (ID). Once the primary forwarding link fails, the protection switching is triggered on the routers adjacent to the failed component. The headers of the packets carried on the failed component are inserted with the ID of a backup table where the failed link was removed. Other routers deliver the affected packets according to the backup table identified in the headers.

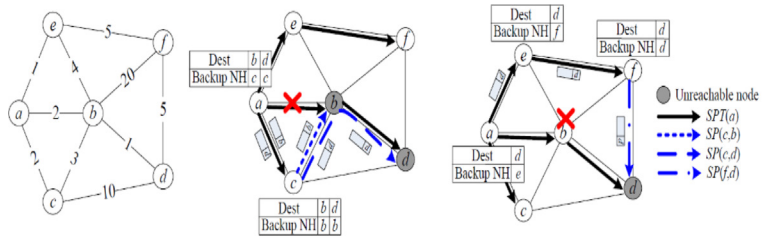
Applying *multi-topology* (MT) to OSPF and Intermediate System-to-Intermediate System protocol (IS-IS) [28] link state routing protocols has recently been standardized [29,30]. Related works on using MT for IP fast local recovery can be found in ref [31]. In the MT routing, each link has a different weight for each topology, whereby different routing classes can be allocated to the corresponding topology. The disadvantage of such approaches is that each router needs to construct and store multiple routing tables.

Efficient SCan for Alternate Paths (ESCAP) and the UAS described here are among the first to consider single-link and single-node failures in IP fast reroute problem [32]. They present two fast rerouting algorithms to handle single-link and single-node failures, respectively. Other related work includes fast path restoration and QoS routing in Multiprotocol Label Switching (MPLS) networks [33–36] and OSPF weight optimization [37]. To enhance MPLS network robustness through resisting failures. The MPLS forwarding is an entirely different methodology for IP network routing.

3 PROPOSED UAS SCHEME

3.1 Notations and definitions

A network topology can be described by a directed graph $G = (\boldsymbol{V},\boldsymbol{E})$, where $\boldsymbol{V} = \{i\}$ denotes the set of nodes; $\boldsymbol{E} = \{l(i,j)\}$ denotes the set of links in the network. According to a link state routing protocol such as OSPF, each node forwards packets to other nodes based on a shortest path tree rooted at itself. The simple topology shown in Fig. 1a is used as an example. Each link is labeled with its cost. Node a constructs its shortest path tree $SPT(a)$ to forward packets to other nodes as shown in Fig. 1b. Let $SP(a,v)$ denote the shortest path from node a to node $v \in \boldsymbol{V}$, $\boldsymbol{E}_{SP(a,v)}$ denote the set of links on $SP(a,v)$, and $\boldsymbol{V}_{SP(a,v)}$ denote the set of nodes on $SP(a,v)$. A router i is said to be an *unaffected router* if router i forwards a packet to destination d using its shortest path without traversing the failed link or node (i.e. the failed link $l(i,j) \notin \boldsymbol{E}_{SP(i,d)}$ and the failed node $y \notin \boldsymbol{V}_{SP(i,d)}$). When the link $l(a,b)$ in Fig. 1b failed, the connection pairs $[a,b]$ and $[a,d]$ are disrupted. These unreachable destinations a and d are called *unreachable nodes*



(a) A simple topology (b) Nodes *a* and *c* use a backup routing table to reroute the packets with destination *b* and *d*. (c) Nodes *a*, *e* and *f* use a backup routing table to reroute the packets with destination *d*.

Figure 1: An example for the healing process of UAS.

of source node a. To further clarify the problem, let $\boldsymbol{V}_{ne1(a)}$ and $\boldsymbol{V}_{ne2(a)}$ denote the sets of one-hop and two-hop neighbors of node a, respectively. For node a, it has $\boldsymbol{V}_{ne1(a)} = \{b,c,e\}$ and $\boldsymbol{V}_{ne2(a)} = \{f,d\}$, where $\boldsymbol{V}_{ne1(a)} \cap \boldsymbol{V}_{ne2(a)} = \varnothing$.

It is of interest to find an *unaffected* alternative neighbor to reroute traffic from a source node a to all the unreachable nodes of the source. This is illustrated in Fig. 1b where the goal is to find unaffected alternative neighbors from the set $\boldsymbol{V}_{ne1(a)}$ to reroute the affected traffic destined for unreachable nodes so as to bypass the failed link $l(a,b)$. If the unaffected alternates from one-hop neighbors cannot be found, two-hop neighbors are used instead. The proposed UAS will guarantee the loop-free routing property during the entire healing process.

3.2 Search alternative neighbors in UAS

The pseudocode of the proposed UAS scheme is shown in Fig. 2. Node x is the performing router; $l(x,y)$ and y denote the failed link and failed node, respectively. Lines 7–22 define the process of healing a single-link failure, whereas lines 23–38 define the healing process of single-node failure. Initially, the backup routing table for protecting link (BRT^L) and backup routing table for protecting node (BRT^N) are empty. The structures of BRT^L and BRT^N are three-dimensional matrices. The first dimension stores the last node, the second dimension stores the destination, the third dimension stores the performing node, and the entry stores the next hop. After the UAS terminates, the computed BRT^L and BRT^N are used to build backup routing table for each router. When there is no confusion, both BRT^L and BRT^N are referred to as BRTs.

For each assumed failed link (or node), the performing node x first checks whether the link (or node) is located on SPT(x). If the link or node is located on SPT(x) and node x is adjacent to the failed link/node, node x seeks an unaffected alternate node from one-hop neighbors for each unreachable destinations included in SPT(x)y. (SPT(x)y is the sub-tree rooted at node y on SPT(x)). If an unaffected alternate node can be found, the unaffected alternate node is inserted into BRT.

If there exists no unaffected alternate node among one-hop neighbors for an unreachable destination, the healing process searches for a two-hop unaffected neighbor, as described by lines 14–21. If a two-hop unaffected alternative neighbor z can be found, a node k within the one-hop neighbors of node x is selected so that node x can reroute the affected traffic to the two-hop unaffected alternative neighbor z via node k. Then, node k and node z are inserted into BRTs of node x and node k, as shown by lines 19–20. The complexity of UAS is $O(|V_{ne}{}^2||N|^2)$ for link or node protection because all nodes (= $|N|$) need to check all its two-hop neighbors (= $|V_{ne}{}^2|$) for protecting all unreachable nodes (= $|N|$ in the worst case).

To illustrate the one-hop and two-hop searching procedures, the example in Fig. 1 is re-visited. Suppose the reached node is a in Fig. 1b. To search for an unaffected alternative neighbor from set $V_{ne1(a)}$ for rerouting the packets destined to the unreachable nodes b and d in case link l(a,b). Node a checks nodes in $V_{ne1(a)}$ to see whether there exists a node whose shortest path can connect to the unreachable nodes b and d without traversing the failed link l(a,b). The rerouted paths are shown in Fig. 1b. Nodes a and c build backup routing tables according to the searching results. In backup routing table, the first row stores the unreachable destinations (Dest) and the second row stores the backup Next Hops (NH). The two-hop searching procedure is triggered only if there exists no unaffected alternate in one-hop neighbors.

3.3 Packet forwarding procedure

When a node s receives a packet for destination d, node s checks whether its primary next hop failed. If the primary next hop failed, node s uses the backup next hop to forward this

Algorithm UAS:

```
1.  Given:
2.    x: performing node.
3.    failed link l(x,y) or failed node y.
4.    SPT(a)_b: the subtree rooted at node b on SPT(a).
5.    BRT^L(:,:,:):=∅. /* BRT^L(last node, destination, performing node)= next hop*/
6.    BRT^N(:,:,:):=∅. /* BRT^N(last node, destination, performing node)= next hop*/
/* for protecting single link failure */
7.  begin
8.    if link l(x,y)∈SPT(x) /*the failure affects |SPT(x)*/
9.      for each node d∈SPT(x)_y do /*unreachable node d*/
10.       for each neighbor z∈V_nei(x)\{y} do
11.         if link l(x,y)∉SP(z,d)
12.           BRT^L(-,d,x):= z;
13.         break;
14.     for each node d∈SPT(x)_y do
15.       if BRT^L(-,d,x)=∅ /*if there is no unaffected router in the one-hop neighbors*/
16.         for each neighbor z∈V_ne2(x)\{SPT(x)_y} do
17.           if link l(x,y)∉SP(z,d)
18.             find a node k∈V_nei(x)\{y} and z∈V_nei(k);
19.             BRT^L(-,d,x):= k;
20.             BRT^L(x,d,k):= z;
21.           break;
22.   end.
/* for protecting single node failure */
23.   begin
24.     if node y∈SPT(x) & y∈V_nei(x)
25.       for each node d∈SPT(x)_y\{y} do /*unreachable node d*/
26.         for each neighbor z∈V_nei(x)\{y} do
27.           if node y∉SP(z,d) /* node y should be protected*/
28.             BRT^N(-,d,x):= z; /*insert backup next hop for backup routing table*/
29.           break;
30.       for each node d∈SPT(x)_y\{y} do
31.         if BRT^N(-,d,x)=∅ /*if there is no unaffected router in the one-hop neighbors*/
32.           for each neighbor z∈V_ne2(x)\{SPT(x)_y} do
33.             if node y∉SP(z,d)
34.               find a node k∈V_nei(x)\{y} and z∈V_nei(k);
35.               BRT^N(-,d,x):= k; /*insert backup next hop for backup routing table*/
36.               BRT^N(x,d,k):= z; /*insert backup next hop for backup routing table*/
37.             break;
38.   end.
```

Figure 2: Pseudocode for UAS algorithm.

packet if the backup next hop exists; otherwise, node *s* drops the packet. If the packet is rerouted by backup next hop, node *s* inserts 1-bit 'tag' into the header of this packet, which tells routers on the backup path to use the backup next hop to forward the packet. If the primary next hop did not fail, node *s* checks whether the packet is marked by 1-bit 'tag'. Node *s* uses the primary next hop to forward this packet if the packet is not marked with 'tag'. If the packet is marked with 'tag', node *s* uses the backup next hop to forward this

packet if the backup next hop exists; node *s* uses the primary next hop to forward the packet if no backup next hop exists.

3.4 Loop-free property of UAS scheme

Theorem: The UAS scheme guarantees that the loop-free routing property always holds during the entire healing process.

***Proof*:** The theorem is proven by contradiction. For the single-link failure case, suppose that the link $l(a,b)$ fails and node d becomes an unreachable node. Suppose that $l(a,b) \in \boldsymbol{E}_{SP(a,d)}$ and node a reroutes affected packets (i.e. the packets carried on the failed component in the normal state) to destination d via an unaffected router x, whereby a loop is formed. In other words, the packets sent from node a to destination d via neighbor x will finally return to node a. Because node x follows its primary routing table (i.e. shortest path) to forward packets to node d, the only possible loop is that node $a \in \boldsymbol{V}_{sp(x,d)}$. Because node x is not aware that link $l(a,b)$ has failed, $\boldsymbol{E}_{sp(a,d)} \subset \boldsymbol{E}_{sp(x,d)}$, and it can be concluded that $\boldsymbol{E}_{SP(x,d)}$ will include the failed link $l(a,b)$. This contradicts the definition of *unaffected router*. Hence, the use of one-hop unaffected router guarantees the loop-free property. The proof for the case of two-hop neighbors can be obtained straightforwardly by extending the case of one-hop neighbor.

The single-node failure case can be proved by following similar procedures and is therefore omitted.

3.5 Traffic dispersion of UAS scheme

In UAS, each node searches one-hop and two-hop unaffected neighbors to reroute the affected traffic flows to reach the unreachable nodes. These affected traffic flows, in fact, are dispersed naturally. As an example, suppose that the unaffected neighbor *e* is used to protect unreachable node *b*, whereas unaffected neighbors *d* and *c* are used to protect unreachable nodes *a* and *y*, respectively.

When link $l(x,y)$ fails, the traffic carried on the failed link $l(x,y)$ is dispersed to the three rerouted paths leading to three respective unreachable destinations via UAS. In addition, the most balanced unaffected neighbor can be selected to reroute the affected traffic and in this way, avoid traffic congestion if there exits more than one unaffected neighbors to protect an unreachable node.

4 PROBLEM FORMULATION

4.1 Assumptions and notations

The problem of maximizing the network survivability with the usage of UAS is formulated as a Mixed Integer Non-linear Program (MINP). The problem of maximizing network survivability is also formulated for two well-known IP fast local recovery schemes, LFA and U-Turn, as MINP. The notations used in the definition of MINP are listed below.

Notations

$\boldsymbol{L}$: Set of network links.

$\boldsymbol{N}$: Set of network nodes.

$\boldsymbol{P}_{(n,d)}$: Set of candidate paths for pair $[n, d]$, and $n \neq d \in \boldsymbol{N}$. Every candidate path does not traverse any node more than once.

$\boldsymbol{L}_n^{out}$: Set of outgoing links of node n.
$\boldsymbol{V}_{ne1(n)}$: Set of one-hop neighbors of node n.
$\boldsymbol{V}_{ne2(n)}$: Set of two-hop neighbors of node n.
$e(l)$: The end node of link l.
ξ: A real number, $0 < \xi < 1$.
M: A large enough number.
y_n^d: = 1, if node n has a backup alternate node to reroute affected packets for destination d; = 0, otherwise.
$\varphi_{n,v}^d$: = 1, if node n has a backup alternate node v to reroute affected packets for destination d; = 0, otherwise.
z_l^d = 1, if link l (or link $l(u,v)$) is selected to be a part of the destination tree rooted at node d; = 0, otherwise.
w_l: Weight of link l for shortest path routing.
x_p: = 1, if path p is used to be a working path; = 0, otherwise.
δ_{pl}: $\delta_{pl} = 1$, if link l is used on path p; $\delta_{pl} = 0$, otherwise.

4.2 Objective function

Given a physical network topology, together with the demand volume which limits the cost of the link for all source–destination (SD) pairs, the problem is to determine weights of all links on working-path routing that maximize the capability of survivability in a network. The objective is formulated as follows.

$$\text{Max} \sum_{d \neq n \in N} y_n^d. \tag{1}$$

The goal here is to encourage the construction of a backup alternate node for each pair $[n,d]$ of source and destination. Three types of the constraints are considered: Working-path routing constraint, destination tree constraint, and weight assignment constraint. Working-path routing constraint refers that for each pair $[n,d]$ node n needs to choose a working path to deliver packets for each destination d in the normal (non-failure) state:

$$\sum_{p \in \boldsymbol{P}_{(n,d)}} x_p = 1, \forall n \in \boldsymbol{N} \setminus \{d\}, d \in \boldsymbol{N}. \tag{2}$$

Destination tree constraint can be defined by the following equations

$$\sum_{l \in \boldsymbol{L}} z_l^d = |\boldsymbol{N}| - 1, \forall d \in \boldsymbol{N}; \tag{3}$$

$$\sum_{p \in \boldsymbol{P}_{(n,d)}} x_p \delta_{pl} \leq M z_l^d, \forall n \in \boldsymbol{N} \setminus \{d\}, d \in \boldsymbol{N}, l \in \boldsymbol{L}; \tag{4}$$

$$z_l^d = 0 \text{ or } 1, \forall l \in \boldsymbol{L}, d \in \boldsymbol{N}. \tag{5}$$

Equation (3) allows the construction of destination trees rooted at each destination d for IP routing. The determined working paths under working-path routing constraint use link l only if $z_l^d = 1$.

For a network running OSPF routing protocol, each working path follows shortest path routing according to the assigned weight on each link. Thus, the weight assignment constraint is considered for each link by the following equations.

$$\sum_{p \in \boldsymbol{P}_{(n,d)}} \sum_{l \in \boldsymbol{L}} x_p \delta_{pl} w_l \leq \sum_{l \in \boldsymbol{L}} a_l \delta_{ql}, \forall q \in \boldsymbol{P}_{(n,d)}, n \in \boldsymbol{N} \setminus \{d\}, d \in \boldsymbol{N}, q \neq p \tag{6}$$

$$w_l \in integer, \forall l \in \boldsymbol{L} \tag{7}$$

$$1 \leq w_l \leq 2^{16} - 1, \forall l \in \boldsymbol{L} \tag{8}$$

The left-hand side of eqn (6) is the length of selected working path. The right-hand side of eqn (6) is the length of a candidate path. By eqn (6), each working path for [*n*,*d*] pair follows the shortest path routing.

1.1 Maximization of the network survivability problem under a network running UAS

The goal of UAS is to find an unaffected alternate node within two-hop neighbors to bypass the failure, which can be derived as

$$\sum_{v \in \boldsymbol{V}_{ne2(n)} \setminus e(l)} \sum_{p \in \boldsymbol{P}_{(v,d)}} (1 - x_p \delta_{pl}) z_l^d \geq y_n^d, \forall l \in \boldsymbol{L}_n^{out}, n \in \boldsymbol{N} \setminus \{d\}, \boldsymbol{d} \in \boldsymbol{N} \tag{9}$$

The right-hand side of eqn (9) (i.e. y_n^d) can be 1 only if $z_l^d = 1$; otherwise, the value of right hand side of eqn (9) is 0. In eqn (9), y_n^d represents the total number of working paths, which do not use link *l*, where the source node *v* of those working paths belong to $\boldsymbol{V}_{ne2(n)} \setminus e(l)$. Therefore, the optimization problem of maximizing the network survivability under a network running UAS can be formulated by the objective function (1) subject to eqns (2)–(9).

5 SARWA HEURISTIC

Simulated annealing (SA) [38] is a probabilistic algorithm to approximate the global optimal solution of a given objective function in a large search space. The SA uses a "temperature" to control the probability of accepting a disadvantageous solution whereby the optimal solution can be approximately approached when the temperature becomes sufficiently cool.

A SA-based heuristic scheme, called SARWA algorithm, is developed to solve routing and link weight assignment problem. The algorithm is depicted by the pseudocode in Fig. 3. Its goal is to determine a set of link weights for working path routing and maximize network survivability at the same time. The proposed SARWA scheme can be applied in a network running any kind of existing IP local recovery schemes, including UAS. In addition to UAS, the LFA and U-Turn described by the pseudocode are embedded in SARWA and their performances are discussed in Section 6.

In the pseudocode, the main operation of simulated annealing is the nested loop from line 7 to line 35. The outer *for* loop controls the value of temperature T, whereas the inner *while* loop repeatedly selects a link and tunes its weight so as to minimize the value of non-protection. The non-protection is defined as the number of working paths without protection. A path said to be *protected* if when any of the node or link on the path fails, the path can be rerouted to destination without using the failed link/node node and incurring loops.

Algorithm SARWA:

```
1.  begin
2.    T=T0; /*initial temperature*/
3.    Ak:= A*;
4.    determine working path for each od-pair by using Dijkstra(Ak) algorithm;
5.    calculate non-protection* based on these computed working paths;
6.    non-protection0 := non-protection*;
7.    for(iter_temp=1; iter_temp≤ MAXIterTemp; iter_temp:= iter_temp+1)
8.      k:=1;
9.      do
10.       Ak:= Ak-1;
11.       t:=random_integer(-ρ,ρ);
12.       l:=random_integer(1,L);
13.       Ak={a1, a2, ..., al + t, ..., aL}; /* update the l-th component of link weight vector Ak */
14.       determine working path routing using Dijkstra(Ak) algorithm;
15.       calculate non-protectionk based on embedded fast recovery approach;
16.       if ( non-protectionk = 0)
17.         non-protection*:= 0;
18.         A*:= Ak ;
19.         return (non-protection*, A*);
20.       else
21.         ω = non-protectionk − non-protectionk-1 ;
22.         if (ω ≤ 0)
23.           if (non-protectionk < non-protection*)
24.             A*:=Ak ;
25.             non-protection*:= non-protectionk ;
26.           end if
27.         else if ( random(0,1) > exp(-ω/T) )
28.           Ak := Ak-1 ;
29.           non-protectionk:= non-protectionk-1;
30.         end if
31.       end if
32.       k=k+1;
33.     while (k ≤ MaxIteration);
34.     T:= T×ε;
35.   end for
36.   return ( non-protection*, A*);
37. end.
```

Figure 3: Pseudocode for (SARWA) algorithm.

The computation of non-protection depends on the IP fast local recovery scheme used by the network.

$\mathbf{A}_k$ is the link weight vector. Initially, the set T_0 is assigned to T and $\mathbf{A}^*$ is assigned to $\mathbf{A}_0$. The working paths and non-protection are then computed at lines 4–5. In each iteration of the inner *while* loop (lines 9–33), a link is selected randomly; its weight is tuned; and the Dijkstra algorithm is applied to calculate the shortest paths (i.e. working paths). The non-protection is computed by the fast recovery scheme installed by the network.

If every working path can be protected, the current $\mathbf{A}^*$ and non-protection* are set to $\mathbf{A}_k$ and 0, respectively. The algorithm returns both of them. Otherwise, the algorithm computes

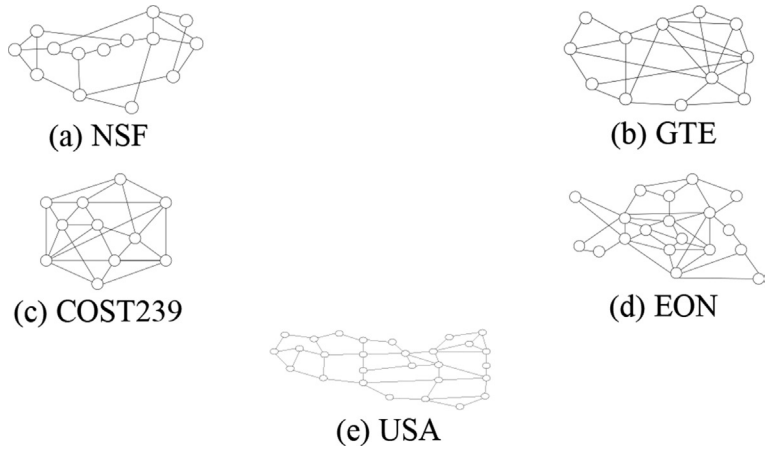


Figure 4: Benchmark networks for performance evaluation.

the difference ω between non-protection (k) and non-protection ($k-1$). If ω is less than or equal to 0 and non-protection (k) is less than non-protection*, the $\mathbf{A}^*$ and non-protection* are updated by $\mathbf{A}_k$ and non-protection$_k$, respectively (see lines 22–26). Otherwise, the algorithm computes the exponential of $-\omega/T$ and generates a random number within (0,1). If the exponential of $-\omega/T$ is less than the generated random number, the $\mathbf{A}_k$ and non-protection (k) are updated by $\mathbf{A}_{k-1}$ and non-protection ($k-1$), respectively (lines 27–30). The outer *for* loop is terminated when $\text{iter}_{\text{temp}}$ is greater than MAXIterTemp, whereas the inner *do* loop is stopped when k is greater than MaxIteration. The temperature T is iteratively multiplied with a value ε ($0 < \varepsilon < 1$) whenever the inner *do* loop is finished. When SARWA scheme terminates, it returns a set of link weights, working routing paths, and the current minimum value of non-protection.

6 PERFORMANCE EVALUATION

This section presents a simulation experiment designed to compute the performance of fast local recovery UUAS, LFA, and U-Turn schemes in benchmark networks shown in Fig. 4 and link weights are computed by SARWA scheme. Our simulations were implemented in C program language.

The recovery schemes are compared according to five key performance metrics: number of non-protected paths, non-protection ratio, convergence behavior, average of necessary entries in a backup routing table, and maximum link load. The two most frequently encountered failures: single-link failures and single-node failures [39,40,41], were considered in our simulations. Specifically, the link weights are first set to be no greater than $2^{16}-1$ (due to the limition of the test hardware) with uniform distribution and performed LFA, U-Turn, and UAS for each of the benchmark networks. Furthermore, the SARWA scheme is applied to maximize the performance of LFA, U-Turn, and UAS. The combined schemes are called LFA-SARWA, U-Turn-SARWA, and UAS-SARWA, respectively. In SARWA, the settings are $\mathbf{A_0} = \{a_l = 100 \mid \forall l \in \boldsymbol{L}\}$, T_0=10, ε = 0.8, MAXIterTemp = 10 and MaxIteration = 10000. The performance results for each single-link or node-failure case presented below is obtained via 100 trials.

6.1 Number of non-protected paths

Tables 1 and 2 show the number of non-protected paths for the different network topologies running LFA, U-Turn, UAS, LFA-SARWA, U-Turn-SARWA, and UAS-SARWA schemes. Figure 5 depicts the corresponding histograms for various recovery schemes and different network topologies listed in Tables 1 and 2.

The results show that UAS scheme outperforms other schemes and SARWA can efficiently reduce the number of non-protected paths by installing the fast recovery schemes. In particular, in a case of single-link failure, U-Turn-SARWA, and UAS-SARWA can reach 100% survivability. Moreover, by performing UAS-SARWA to protect single-node failures, only 17 paths and four paths in the USA and EON networks cannot be protected, respectively. It can be observed that UAS has excellent performance for node protection.

The non-protection ratio is the ratio of the total number of non-protected paths to the total number of disrupted connections. The non-protection ratios for two single-link and single-node failure cases are shown in Tables 3 and 4, respectively. Figure 6 depicts the histogram from Tables 3 and 4. As Fig. 6 shows that in comparison to LFA, embedding proposed SARWA scheme in LFA can reduce non-protection ratio by 0.312 for single-link failure cases, and 0.401 for single-node failure cases. While the non-protection ratios of U-Turn and UAS were also reduced to almost 0 through SARWA.

6.2 Convergence behavior

In this experiment, the rates of convergence for LFA-SARWA, U-Turn-SARWA, and UAS-SARWA were observed. The simulation results are shown in Fig. 7. It can be seen from Fig. 7

Table 1: Number of non-protected paths with fast reroute scheme for single-link failures.

	NSF	USA	COST239	EON	GTE
LFA	209.5	1048.8	13.3	176.1	54.6
U-Turn	28.9	161	1	20.5	12.8
UAS	27.1	137	0	6.7	9
LFA-SARWA	78	250	0	11	0
U-Turn-SARWA	0	0	0	0	0
UAS-SARWA	0	0	0	0	0

Table 2: Number of non-protected paths with fast reroute schemes for single-node failures.

	NSF	USA	COST239	EON	GTE
LFA	160.3	1380.9	22.8	284.8	67.7
U-Turn	70.3	800.6	8	164.3	27.2
UAS	9.7	308.1	0	75.9	4.8
LFA-SARWA	64	402	0	58	2
U-Turn-SARWA	9	74	0	4	0
UAS-SARWA	0	17	0	4	0

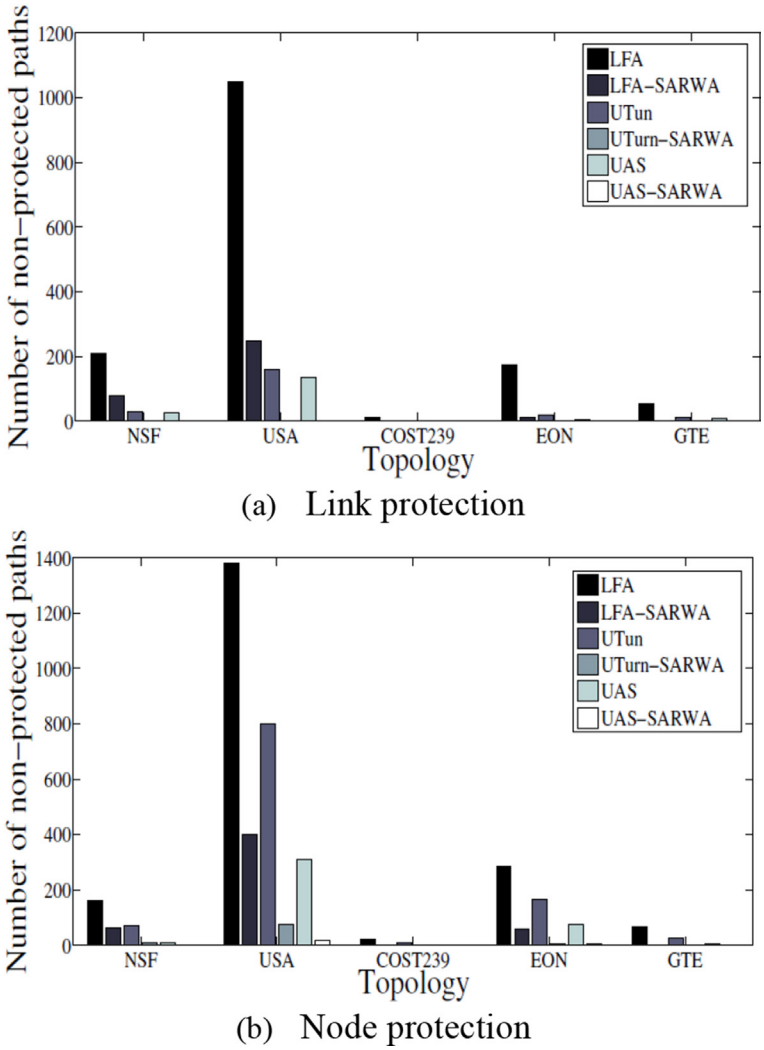


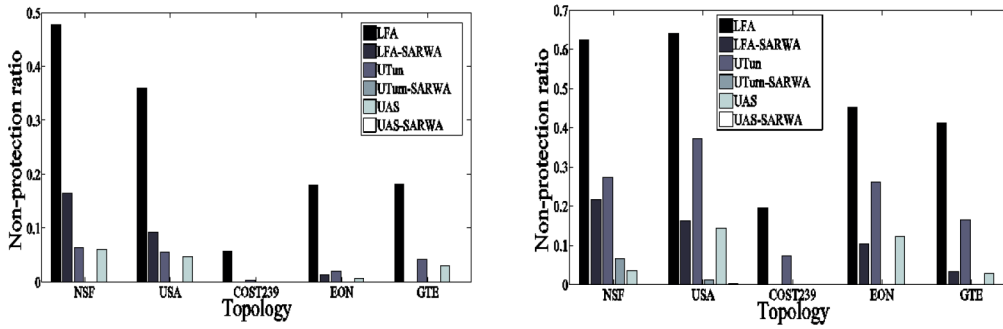
Figure 5: Number of non-protected paths for (a) link protection under single-link failures; and (b) node protection under single-node failures.

Table 3: Non-protection ratio with fast reroute schemes for single-link failures.

	NSF	USA	COST239	EON	GTE
LFA	0.477	0.359	0.057	0.18	0.182
U-Turn	0.064	0.054	0.0036	0.021	0.042
UAS	0.06	0.047	0	0.0071	0.029
LFA-SARWA	0.165	0.092	0	0.013	0
U-Turn-SARWA	0	0	0	0	0
UAS-SARWA	0	0	0	0	0

Table 4: Non-protection ratio with fast reroute schemes for single-node failures.

	NSF	USA	COST239	EON	GTE
LFA	0.623	0.641	0.194	0.451	0.41
U-Turn	0.271	0.372	0.071	0.261	0.163
UAS	0.034	0.143	0	0.121	0.027
LFA-SARWA	0.222	0.179	0	0.115	0.018
U-Turn-SARWA	0.036	0.031	0	0.007	0
UAS-SARWA	0	0.0068	0	0.0062	0



(a) Link protection (b) Node protection

Figure 6: Non-protection ratio for (a) link protection under single-link failures; and (b) node protection under single-node failures.

that with the increasing number of iterations, the number of non-protected paths and non-protection ratio were reduced. For link protection, U-Turn-SARWA and UAS-SARWA converge within 1000 iterations. It is hypothesized that the number of non-protected paths for LFA-SARWA in NSF network can be further reduced if the number of performed iterations is increased. In node protection, UAS-SARWA can be reduced to almost 0 within 100K iterations. Similarly, the number of non-protected paths and non-protection ratio for LFA-SARWA and U-Turn-SARWA can be further reduced with increasing the number of iterations. In addition, the convergence rate of each scheme in USA network is faster than in NSF network due to the fact that the average network degree of USA (= 3.2143) is larger than that of NSF (= 3). The larger average network degree implies higher chances of finding alternate neighbor for LFA, U-Turn, and UAS.

6.3 Number of entries in the backup routing table

The UAS-SARWA was also simulated on benchmark networks to observe the average number of entries in a backup routing table. The results are shown in Fig. 8. It was discovered that UAS-SARWA only needs 1–3 backup routing entries in each network. The UAS-SARWA needs the highest number of entries in NSF network and the least number of entries for EON network. The results indicate that proposed UAS-SARWA only needs few extra backup routing entries while achieving high network survivability.

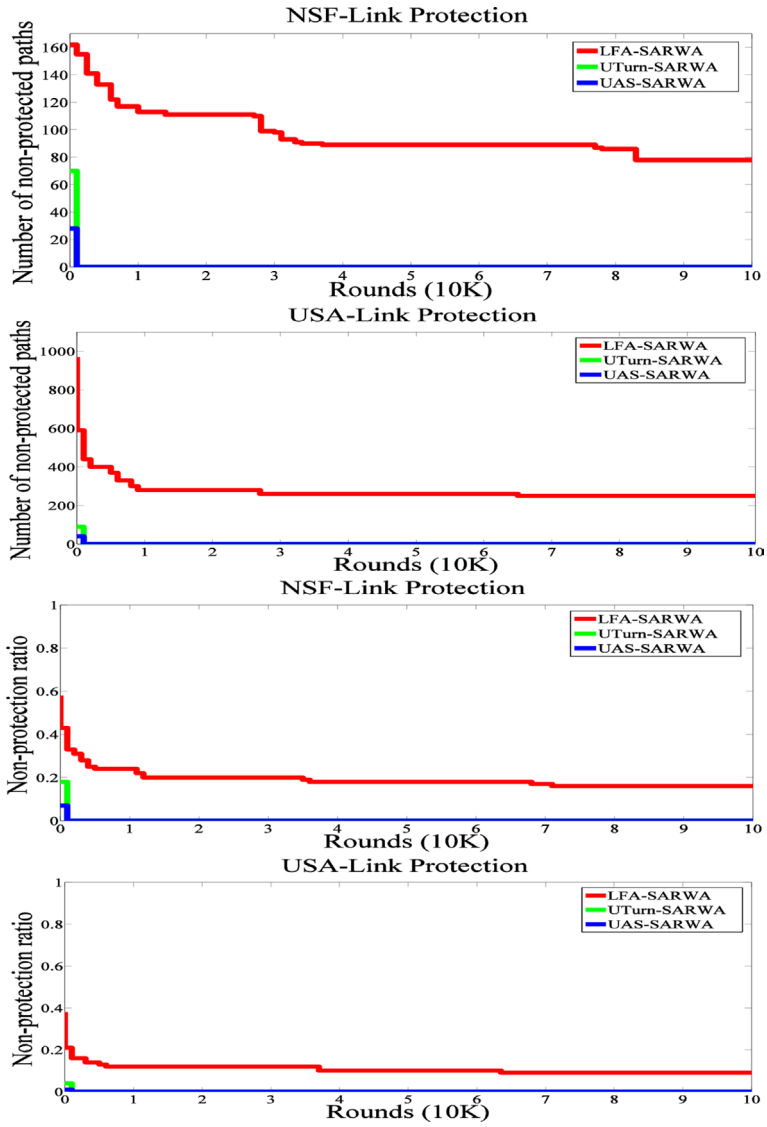


Figure 7: The healing convergence behaviors of using LFA-SARWA, U-Turn-SARWA, and UAS-SARWA, (a) under link protection and (b) under node protection.

6.4 Maximum link load

The results in Sections 6.2 and 6.3 show that UAS can achieve 100% survivability for network COST239 by SARWA scheme regardless of choices of the link weights.

To evaluate how well the proposed UAS can balance loads, the objective of SARWA is changed into minimizing the maximum load on the most congested link (referred to as the maximum link load for short) in each failure state. The non-protection function of pseudocodes in Fig. 3 is changed into MaxLinkLoad function so as to find the maximum

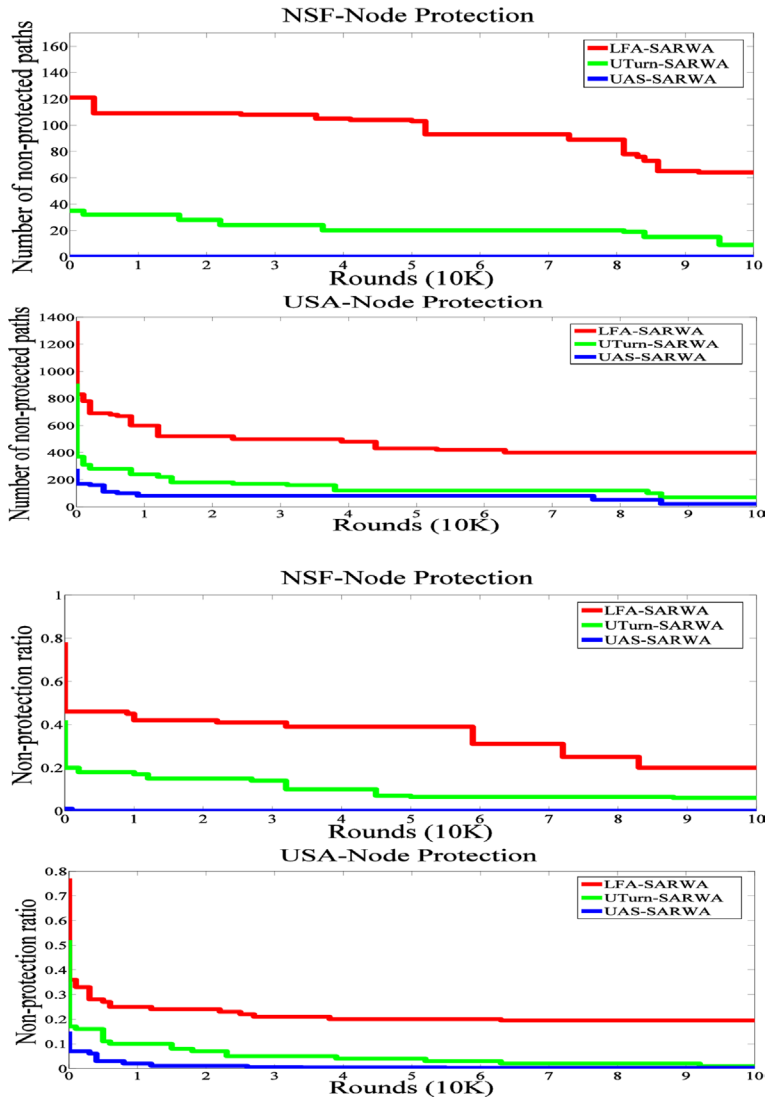


Figure 7: (*Continued*).

link load in each failure state. The main procedure was also to repeatedly pick a link randomly and tune its weight. If new link weights result in lower maximum link load, the new link weights and maximum link load are then stored. The optimal minimum–maximum link load and the corresponding link weights are generated as the output of the scheme when the maximum number of iterations is reached.

The settings include outer stop criterion MAXIterTemp = 10, the inner stop criterion MaxIteration = 10000, and the traffic demand between any two nodes as 10 Mbps. The simulation results are shown in Fig. 9. The *x*-axis represents the failed link/node index, where the non-failure states are shown at the link/node index = 0. The maximum link load can be efficiently reduced by more than 50% for each single-link or single-node failure event. The

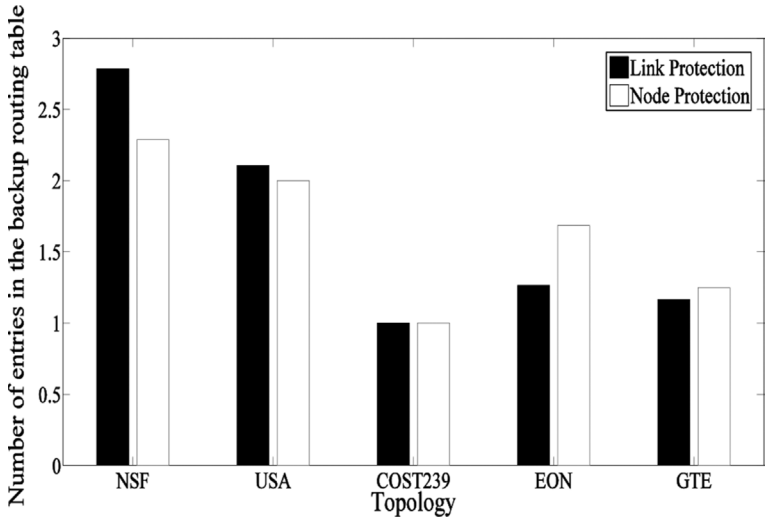


Figure 8: Average of necessary entries in a backup routing table for single-link and single-node failures.

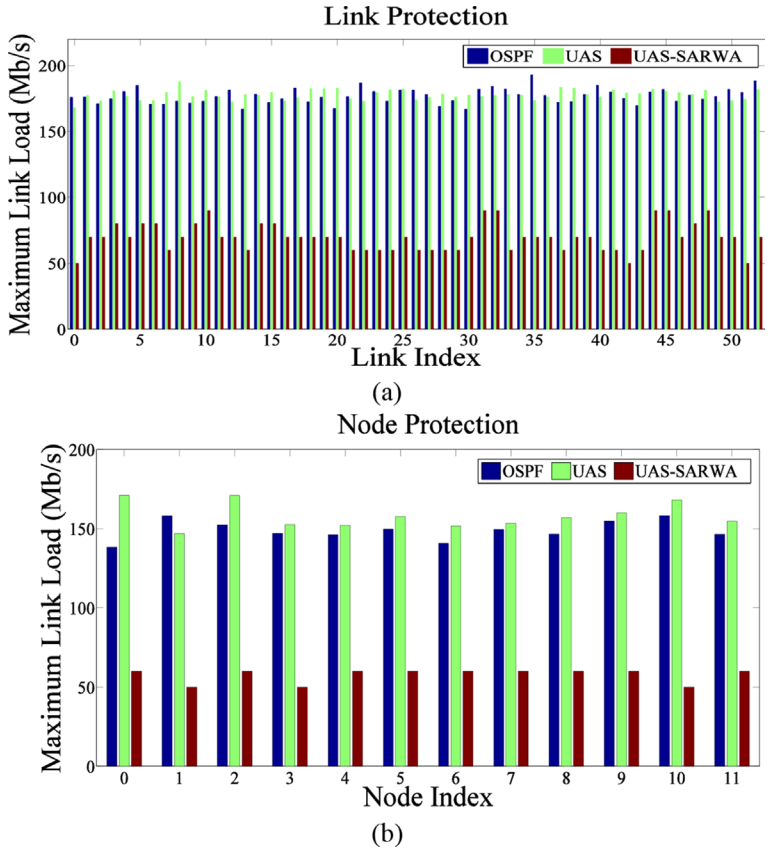


Figure 9: Maximum link load for (a) link protection under single-link failures; and (b) node protection under single-node failures.

results demonstrate that the proposed UAS-SARWA efficiently balance the traffic loads in non-failure state, single-link failure state, and single-node failure state.

7 CONCLUSION

This article described a simple and high-survivability IP protection scheme, called UAS. The scheme computes a backup routing table for each router when there is no node and link failure, and delivers the packets via the alternate routing table to avoid the failed link or node when there is a failure. The UAS has been verified to hold the loop-free routing property at any time during the entire healing process. The article also describes a SARWA scheme to determine link weight for working paths routing and to maximize the survivability of a network that uses IP fast local recovery. SARWA has been applied in UAS, as well as LFA and U-Turn, two well-known IP fast reroutes. Simulation results demonstrate that the survivability of LFA can be enhanced from 50% to 82% for link protection and from 40% to 78% for node protection through SARWA. The U-Turn and UAS can reach approximately 100% survivability for both single-link and single-node failure cases through SARWA. In particular, UAS shows very good performance in node protection cases. The simulation results also indicate that SARWA can efficiently balance traffic loads. The maximum load on the most congested link can be reduced more than 50% for each single-link or single-node failure. The future works related to UAS and SARWA include the double-link/SRLG failures and the improvements in the random search procedure of SARWA.

REFERENCES

[1] Basu, A. & Riecke, J.G., Stability issues in OSPF routing. *Proceedings of ACM SIGCOMM*, pp. 225–236, August 2001. doi: http://dx.doi.org/10.1145/383059.383077

[2] Watson, D., Jahanian, F. & Labovitz, C., Experiences with monitoring OSPF on a regional service provider network. *Proceedings of IEEE ICDCS*, pp. 204–213, 2003.

[3] Moy, J., OSPF Version 2, IETF RFC 2328, April 1998. doi: http://dx.doi.org/10.1109/icdcs.2003.1203467

[4] Goyal, M., Ramakrishnan, K.K. & Feng, W.-C., Achieving faster failure detection in OSPF networks. *Proceedings of IEEE ICC*, pp.296–300, May 2003. doi: http://dx.doi.org/10.1109/icc.2003.1204188

[5] Francois, P., Filsfils, C., Evans, J. & Bonaventure, O., Achieving sub-second IGP convergence in large IP networks. *ACM SIGCOMM Computer Communication Review*, **35(2)**, pp. 35–44, 2005. doi: http://dx.doi.org/10.1145/1070873.1070877

[6] Alattinoglu, C., Jacobson, V. & Yu, H., Towards milli-second IGP convergence, IETF Internet Draft, draft-alaettinoglu-ISISconvergence-00.txt, November 2000.

[7] Alattinoglu, C. & Casner, S., ISIS routing on the Qwest backbone: a recipe for subsecond ISIS convergence. Presented at the *NANOG Meeting*, Miami, FL, 2002.

[8] Eramo, V., Listanti, M. & Cianfrani, A., Design and evaluation of a new multi-path incremental routing algorithm on software routers. *IEEE/ACM Transactions on Network and Service Management*, **4(4)**, pp. 188–203, 2008. doi: http://dx.doi.org/10.1109/tnsm.2009.041101

[9] Francois, P. & Bonaventure, O., Avoiding transient loops during the convergence of link-state routing protocols. *IEEE/ACM Transactions on Networking*, **15(6)**, pp. 1280–1292, 2007. doi: http://dx.doi.org/10.1109/tnet.2007.902686

[10] Fu, J., Sjödin, P. & Karlsson, G., Loop-free updates of forwarding tables. *IEEE/ACM Transactions on Network and Service Management*, **5(1)**, pp. 22–35, 2008. doi: http://dx.doi.org/10.1109/tnsm.2008.080103

[11] Hopps, C. Analysis of an equal-cost multi-path algorithm. IETF RFC 2992, 2000.
[12] Atlas, A. & Zinin, A., Basic specification for IP fast reroute: loop-free alternates, IETF Internet Draft, draft-ietf-rtgwg-ipfrr-spec-base-04.txt, 2005.
[13] Atlas, A. U-turn alternates for IP/LDP fast-reroute, IETF Internet Draft, draft-atlas-ip-local-protect-uturn-03.txt, 2006.
[14] Raj, A. & Ibe, O.C. A survey of IP and multiprotocol label switching fast reroute schemes. *Computer Networks*, **51(8)**, pp. 1882–1907, 2007. doi: http://dx.doi.org/10.1016/j.comnet.2006.09.010
[15] Gjoka, M., Ram, V. & Yang, X., Evaluation of IP fast reroute proposals. *Proceedings of COMSWARE*, pp. 1–8, Jan. 2007. doi: http://dx.doi.org/10.1109/comswa.2007.382443
[16] Bryant, S., Filsfils, C., Previdi, S. & Shand, M., IP fast reroute using tunnels, IETF Internet Draft, draft-bryantipfrr-tunnels-02.txt, April 2005.
[17] Bryant, S., Shand, M. & Previdi, S., IP fast reroute using not-via addresses, IETF Internet Draft, draft-bryant-shand-IPFRR-notviaaddresses-01.txt, 2005.
[18] Kini, S., Ramasubramanian, S., Kvalbein, A. & Hansen, A.F., Fast recovery from dual-link or single-node failures in IP networks using tunneling. *IEEE/ACM Transactions on Networking*, **18(6)**, pp. 1988–1999, December 2010. doi: http://dx.doi.org/10.1109/tnet.2010.2055887
[19] Kvalbein, A., Hansen, A.F., Čičić, T., Gjessing, S. & Lysne, O., Fast IP network recovery using multiple routing configurations. *Proceedings of IEEE INFOCOM*, April 2006. doi: http://dx.doi.org/10.1109/infocom.2006.227
[20] Kvalbein, A., Hansen, A.F., Čičić, T., Gjessing, S. & Lysne, O., Multiple routing configurations for fast IP network recovery. *IEEE/ACM Transactions on Networking*, **17(2)**, pp. 473–486, April 2009. doi: http://dx.doi.org/10.1109/tnet.2008.926507
[21] Čičić, T., Hansen, A.F., Kvalbein, A., Hartmann, M., Martin, R., Menth, M., Gjessing, S. & Lysne, O., Relaxed multiple routing configurations: IP fast reroute for single and correlated failures. *IEEE/ACM Transactions on Network and Service Management*, **6(1)**, pp. 1–14, 2009. doi: http://dx.doi.org/10.1109/tnsm.2009.090301
[22] Nelakuditi, S., Lee, S., Yu, Y., Zhang, Z.-L. & Chuah, C.-N., Fast local rerouting for handling transient link failures. *IEEE/ACM Transactions on Networking*, **15(2)**, pp. 359–372, April 2007. doi: http://dx.doi.org/10.1109/tnet.2007.892851
[23] Menth, M. & Martin, R., Network resilience through multi-topology routing. *Proceedings of the 5th International Workshop on Design of Reliable Communication Networks (DRCN)*, October 2005. doi: http://dx.doi.org/10.1109/drcn.2005.1563878
[24] Kvalbein, A., Čičić, T. & Gjessing, S., Post-failure routing performance with multiple routing configurations. *Proceedings of IEEE INFOCOM*, April 2007. doi: http://dx.doi.org/10.1109/infcom.2007.20
[25] Nelakuditi, S., Lee, S., Yu, Y. & Zhang, Z.-L., Failure insensitive routing for ensuring service availability. *Proceedings of the International Workshop Quality Service (IWQoS)*, pp. 287–304, 2003. doi: http://dx.doi.org/10.1007/3-540-44884-5_16
[26] Lee, S., Yu, Y., Nelakuditi, S., Zhang, Z.-L. & Chuah, C.-N., Proactive versus reactive approaches to failure resilient routing. *Proceedings of IEEE INFOCOM*, pp. 176–186, 2004. doi: http://dx.doi.org/10.1109/infcom.2004.1354492
[27] Zhong, Z., Nelakuditi, S., Yu, Y., Lee, S., Wang, J. & Chuah, C.-N., Failure inferencing based fast rerouting for handling transient link and node failures. *Proceedings of IEEE INFOCOM*, pp. 1–5, April 2006. doi: http://dx.doi.org/10.1109/infcom.2005.1498576
[28] Oran, D., OSI IS-IS intra-domain routing protocol, IETF Request for Comments 1142.

[29] Psenak, P., Mirtorabi, S., Roy, A., Nguen, L. & Pillay-Esnault, P., MTOSPF: Multi topology (MT) routing in OSPF, IETF, RFC4915, June 2007.
[30] Przygienda, T., Shen, N. & Sheth, N., M-ISIS: Multi topology (MT) routing in IS-IS, IETF RFC 5120, February 2008.
[31] Kwong, K.-W., Guérin, R., Shaikh, A. & Tao, S., Balancing performance, robustness and flexibility in routing systems. *IEEE/ACM Transactions on Network and Service Management*, **7(3)**, pp. 186–199, September 2010. doi: http://dx.doi.org/10.1145/1544012.1544022
[32] Xi, K. & Chao, H. J., IP fast rerouting for single-link/node failure recovery. *Fourth International Conference on Broadband Communications, Networks and Systems (BROADNETS)*, pp. 142–151, September 2007. doi: http://dx.doi.org/10.1109/broadnets.2007.4550418
[33] Bejerano, Y., Breitbart, Y., Orda, A., Rastogi, R. & Sprintson, A., Algorithms for computing QoS paths with restoration. *Proceedings of IEEE INFOCOM*, pp. 1435–1445, 2003. doi: http://dx.doi.org/10.1109/infcom.2003.1208979
[34] Wei, C.-Y. & Naraghi-Pour, M., Path restoration with QoS and label constraints in MPLS networks. *Proceedings of IEEE ICC*, pp. 1278–1282, 2004. doi: http://dx.doi.org/10.1109/icc.2004.1312705
[35] Wang, D. & Li, G., Efficient distributed bandwidth management for MPLS fast reroute. *IEEE/ACM Transactions on Networking*, **16(2)**, pp. 486–495, 2008. doi: http://dx.doi.org/10.1109/tnet.2007.900411
[36] Sharma, V., Framework for MPLS-based recovery, IETF Internet Draft, draft-ietf-mpls-recovery-frmwrk-03.txt, January 2002.
[37] Fortz, B. & Thorup, M., Internet traffic engineering by optimizing OSPF weights. *Proceedings of IEEE INFOCOM*, pp. 519–528, 2000. doi: http://dx.doi.org/10.1109/infcom.2000.832225
[38] Kirkpatrick, S., Gelatt, C.D., Jr. & Vecchi, M.P. Optimization by simulated annealing *Science*, **220(4598)**, pp. 671–680, May 1983.
[39] Iannaccone, G., Chuah, C., Mortier, R., Bhattacharyya, S. & Diot, C., Analysis of link failures in an IP backbone. *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, November 2002. doi: http://dx.doi.org/10.1145/637235.637238
[40] Markopoulou, A., Iannaccone, G., Bhattacharyya, S., Chuah, C.-N. & Diot, C., Characterization of failures in an IP backbone network. *Proceedings of IEEE INFOCOM*, March 2004. doi: http://dx.doi.org/10.1109/infcom.2004.1354653
[41] Tseng, P., Chung, W., Lin, K.C., Shih, C. & Chen, L., Physical-layer communication recovery for heterogeneous network. *Disaster Management and Human Health Risk III: Reducing Risk, Improving Outcomes*, WIT Press: *A Coruña*, Spain, 2013. doi: http://dx.doi.org/10.2495/dman130181