National Conference on Advances in Computer Science and Applications with International Journal of Computer Applications (NCACSA 2012)
Proceedings published in International Journal of Computer Applications® (IJCA)

A Network Recovery Scheme for Node or Link Failures using Multiple Routing Configurations

Suresh Babu Panatula
Department of Computer Science and Engineering
Sri Sai Aditya Institute of Science and Technology,
Surampalem, East Godavari District, AP, INDIA

Subbarao Dusari
Department of Computer Science and Engineering
Aditya Engineering College, Surampalem,
East Godavari District, A.P, INDIA

ABSTRACT
Today in communication infrastructure internet takes major role, after the network problems the slow convergence of routing protocols becomes a increasing problem. Here the proposed scheme guarantees recovery in all single failure scenarios, using the single mechanism to handle both link and node failures. MRC is very straight forward and assumes only the destination based hop-by-hop forwarding. It will have all the information of the routers and packet forwarding also takes place. And also show estimate of the traffic demands. MRC is strictly connectionless, and assumes only destination based hop-by-hop forwarding. It can be implemented with only minor changes to existing solutions. In this paper I present MRC, and analyze its performance with respect to scalability, backup path lengths, and load distribution after a failure.

General Terms
Network Recovery Scheme, Node or Link failures, Multiple Routing Configurations

Keywords
Network Recovery, Multiple Routing Configurations

1. INTRODUCTION
In Recent years the Internet has been transformed from a special purpose network to an ubiquitous platform for a wide range of everyday communication services. The demands on Internet reliability and availability have increased accordingly. A disruption of a link in central parts of a network has the potential to affect hundreds of thousands of phone conversations or TCP connections, with obvious adverse effects. The ability to recover from failures has always been a central design goal in the Internet 1. IP networks are intrinsically robust, since IGP routing protocols like OSPF are designed to update the forwarding information based on the changed topology after a failure. This re-convergence assumes full distribution of the new link state to all routers in the network domain. When the new state information is distributed, each router individually calculates new valid routing tables. This network-wide IP re-convergence is a time consuming process, and a link or node failure is typically followed by a period of routing instability. During this period, packets may be dropped due to invalid routes. This phenomenon has been studied in both IGP 2 and BGP context 3, and has an adverse effect on real-time applications 4. Events leading to a re-convergence have been shown to occur frequently. A key problem is that since most network failures are short lived, too rapid triggering of the re-convergence process can cause route flapping and increased network instability 2. The IGP convergence process is slow because it is reactive and global. It reacts to a failure after it has happened, and it involves all the routers in the domain. In this paper we present a new scheme for handling link and node failures in IP networks. Then Multiple Routing Configurations (MRC) is a proactive and local protection mechanism that allows recovery in the range of milliseconds. MRC allows packet forwarding to continue over preconfigured alternative next-hops immediately after the detection of the failure.

2. MRC OVERVIEW
MRC is based on building a small set of backup routing configurations that are used to route recovered traffic on alternate paths after a failure. The backup configurations differ from the normal routing configuration in that link weights are set so as to avoid routing traffic in certain parts of the network. We observe that if all links attached to a node are given sufficiently high link weights, traffic will never be routed through that node. The failure of that node will then only affect traffic that is sourced at or destined for the node itself. Similarly, to exclude a link (or a group of links) from taking part in the routing, we give it infinite weight. The link can then fail without any consequences for the traffic. Our MRC approach is threefold. First, we create a set of backup configurations, so that every network component is excluded from packet forwarding in one configuration.

Second, for each configuration, a standard routing algorithm like OSPF is used to calculate configuration specific shortest paths and create forwarding tables in each router, based on the configurations. The use of a standard routing algorithm guarantees loop-free forwarding within one configuration. Finally, we design a forwarding process that takes advantage of the backup configurations to provide fast recovery from a component failure.

3. GENERATING BACKUP CONFIGURATIONS
In this section, we will first detail the requirements that must be put on the backup configurations used in MRC. Then, we propose an algorithm that can be used to automatically create such configurations. The algorithm will typically be run once at the initial start-up of the network, and each time a node or link is permanently added or removed. We use the notation shown in Table I.

3.1 Configurations Structure
MRC configurations are defined by the network topology, which is the same in all configurations, and the associated link weights, which differ among configurations. We formally represent the network topology as a graph G (N, A), with a set

of nodes and a set of unidirectional links (arcs). MRC is agnostic to the getting of these weights. In the backup configurations, selected links and nodes must not carry any transit traffic. Still, traffic must be able to depart from and reach all operative nodes. These traffic regulations are imposed by assigning high weights to some links in the backup configurations:

Definition: A link $a \in A$ is *isolated* in C_i if $w_i(a) = \infty$.
Definition: A link $a \in A$ is *restricted* in C_i if $w_i(a) = w_r$.

With MRC, restricted and isolated links are always attached to isolated nodes as given by the following rules. For all links

$$(u, v) \in A,$$

$$w_i(u,v) = w_r \Rightarrow (u \in S_i \wedge v \in \overline{S}_i) \vee (v \in S_i \wedge u \in \overline{S}_i)$$
$$w_i(u,v) = \infty \Rightarrow u \in S_i \vee v \in S_i.$$

This means that a restricted link always connects an isolated node to a non-isolated node.

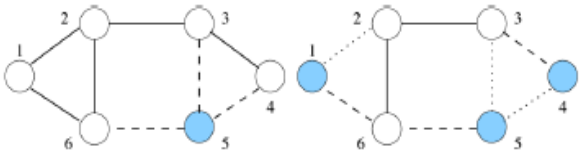

Fig.1. Left: node 5 is isolated (shaded color) by setting a high weight on its entire connected links (stapled). Only traffic to and from the isolated node will use these restricted links. Right: a configuration where nodes 1, 4 and 5, and the links 1–2, 3–5 and 4–5 are isolated (dotted). Importantly, this means that a link is always isolated in the same configuration as at least one of its attached nodes. These two rules are required by the MRC forwarding process described in Section IV in order to give correct forwarding without knowing the root cause of failure.

Definition: A configuration C_i is *valid* if and only if

$$\forall u, v \in N : \mathcal{N}(p_i(u,v)) \setminus (\overline{S}_i \cup \{u, v\}) = \emptyset$$
$$\wedge w_i(p_i(u,v)) < \infty.$$

We observe that all backup configurations retain a characteristic internal structure, in that all isolated nodes are directly connected to a core of nodes connected by links with normal weights:

Definition: A configuration *backbone* $B_i = (S_i, Ai)$ consists of all non-isolated nodes in C_i and all links that are neither isolated nor restricted:

$$a \in A_i \Leftrightarrow w_i(a) \leq w_{\max}.$$

A backbone is connected if all nodes in S_i are connected by paths containing links with normal weights only:

Definition: A backbone B_i is *connected* if and only if

$$\forall u, v \in B_i : a \in \mathcal{A}(p_i(u,v)) \Rightarrow w_i(a) \leq w_{\max}$$

An important invariant in our algorithm for creating backup configurations is that the backbone remains connected. We can show that a backup configuration with a connected backbone is equivalent to a valid backup configuration.

3.2 Algorithm

The number and internal structure of backup configurations in a complete set for a given topology may vary depending on the construction model. If more configurations are created, fewer links and nodes need to be isolated per configuration, giving a richer (more connected) backbone in each configuration. Instead we present a heuristic algorithm that attempts to make all nodes and links in an arbitrary bi-connected topology isolated. Our algorithm takes as input the directed graph and the number of backup configurations that is intended created. If the algorithm terminates successfully, its output is a complete set of valid backup configurations. The algorithm is agnostic to the original link weights, and assigns new link weights only to restricted and isolated links in the backup configurations. For a sufficiently high, the algorithm will always terminate successfully, as will be further discussed in Section III-B-3. We now specify the algorithm in detail, using the notation shown in Table1:

Table: 1

$G = (N, A)$	Graph comprising nodes N and directed links (arcs) A
C_i	The graph with link weights as in configuration i
S_i	The set of isolated nodes in configuration C_i
B_i	The backbone in configuration C_i
$A(u)$	The set of links from node u
(u, v)	The directed link from node u to node v
$p_i(u, v)$	A given shortest path between nodes u and v in C_i
$\mathcal{N}(p)$	The nodes on path p
$\mathcal{A}(p)$	The links on path p
$w_i(u, v)$	The weight of link (u, v) in configuration C_i
$w_i(p)$	The total weight of the links in path p in configuration C_i
w_r	The weight of a restricted link
n	The number of configurations to generate (algorithm input)

Function `isolate`(C_i, u)

```
1  Q_a ← Q_a + (u,v), ∀(u,v) ∈ A(u)
2  while Q_a ≠ ∅ do
3      (u,v) ← first (Q_a)
4      if ∃j : v ∈ S_j then
5          if w_j(u,v) = w_r then
6              if ∃(u,x) ∈ A(u)\(u,v) : w_i(u,x) ≠ ∞ then
7                  w_i(u,v) ← w_i(v,u) ← ∞
8              else
9                  return null
10         else if w_j(u,v) = ∞ and i ≠ j then
11             w_i(u,v) ← w_i(v,u) ← w_r
12     else
13         if ∃(u,x) ∈ A(u)\(u,v) : w_i(u,x) ≠ ∞ then
14             w_i(u,v) ← w_i(v,u) ← ∞
15         else
16             w_i(u,v) ← w_i(v,u) ← w_r
17             Q_a ← v + (Q_a \ v)
18             Q_a ← (v,u)
19 end
20 return C_i
```

Algorithm 1: Creating backup configurations.

```
1  for i ∈ {1...n} do
2      C_i ← (G, w_0)
3      S_i ← ∅
4      B_i ← C_i
5  end
6  Q_n ← N
7  Q_s ← ∅
8  i ← 1
9  while Q_n ≠ ∅ do
10     u ← first(Q_n)
11     j ← i
12     repeat
13         if connected(B_i \ ({u}, A(u))) then
14             C_tmp ← isolate(C_i, u)
15             if C_tmp ≠ null then
16                 C_i ← C_tmp
17                 S_i ← S_i ∪ {u}
18                 B_i ← B_i \ ({u}, A(u))
19         i ← (i mod n) + 1
20     until u ∈ S_i or i=j
21     if u ∉ S_i then
22         Give up and abort
23 end
```

a) Main loop: Initially, backup configurations are created as copies of the normal configuration. A queue of nodes and a queue of links are initiated. The node queue contains all nodes in an arbitrary sequence. The link queue is initially empty, but all links in the network will have to pass through it. Method returns the first item in the queue, removing it from the queue. When a node is attempted isolated in a backup configuration, it is first tested that doing so will not disconnect according to definition

b) Isolating links: Along with, as many as possible of its attached links are isolated. The algorithm runs through the links attached to (lines 2–3 in function). It can be shown that it is an invariant in our algorithm that in line 1, all links in are attached to node. The node in the other end of the link may or may not be isolated in some configuration already (line 4). If it is, we must decide whether the link should be isolated along with (line 7), or if it is already isolated in the configuration where is isolated (line 11).

4. LOCAL FORWARDING PROCESS

Given a sufficiently high, the algorithm presented in Section III will create a complete set of valid backup configurations. Based on these, a standard shortest path algorithm is used in each configuration to calculate configuration specific forwarding tables. In this section, we describe how these forwarding tables are used to avoid a failed component. When a packet reaches a point of failure the node adjacent to the failure, called the detecting node, is responsible for finding a backup configuration where the failed component is isolated

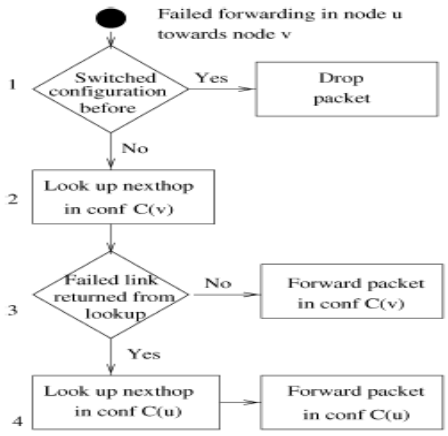

Fig. 2. Packet forwarding state diagram.

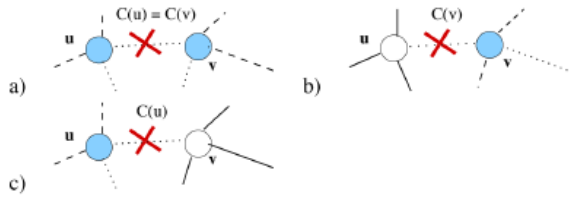

Fig.3. When there is an error in the last hop U->V a packet must be forwarded in the configuration where the connecting link is isolated. The figure shows isolated nodes (shaded color), restricted links (dashed), and isolated links (dotted). In cases (a) and (b), C(u, v)=C(v) and the forwarding will be done in C(V) In case (c) C(u, v)≠C(v), and the forwarding will be done in C(u).

For the routers to make a correct forwarding decision, each packet must carry information about which configuration it belongs to. This information can be either explicit or implicit. An explicit approach could be to use a distinct value in the DSCP field of the IP header to identify the configuration. As we will see shortly, a very limited number of backup configurations are needed to guarantee recovery from all single link or node failures, and hence the number of needed values would be small. A more implicit approach would be to assign a distinct local IP address space for each backup configuration. Each node in the IGP cloud would get a separate address in each configuration. The detecting node could then encapsulate recovered packets and tunnel them shortest path in the selected backup configuration to the egress node. The packets would then be encapsulated at the egress and forwarded from there as normal towards the final destination. The drawback with this method is the additional processing and bandwidth resource usage associated with tunnelling.

5. PERFORMANCE EVALUAION

MRC requires the routers to store additional routing configurations. The amount of state required in the routers is related to the number of such backup configurations. Since routing in a backup configuration is restricted, MRC will potentially give backup paths that are longer than the optimal paths. Longer backup paths will affect the total network load and also the end-to-end delay. Full, global IGP re-convergence determines shortest paths in the network without the failed component. We use its performance as a reference point and evaluate how closely MRC can approach it. It must be noted that MRC yields the shown performance immediately after a failure, while IP re-convergence can take seconds to complete.

5.1 Evaluation Setup

We have implemented the algorithm described in Section III-B and created configurations for a wide range of bi-connected synthetic and real topologies. To explore the effect of network density, the average node degree is 4 or 6 for Waxman topologies and 3.6 for GLP topologies. For all synthetic topologies, the links are given unit weight.

The real topologies are taken from the Rocket fuel topology database. For each topology, we measure the minimum number of backup configurations needed by our algorithm to isolate every node and link in the network. Recall from

Section III-B that our algorithm for creating backup configurations only takes the network topology as input, and is not influenced by the link weights.
Hence, the number of configurations needed is valid irrespective of the link weight settings used. For the Rocket fuel topologies, we also measure the number of configurations needed if we exclude the nodes that can be covered by Loop-Free Alternates (LFA) . LFA is a cheaper fast reroute technique that exploits the fact that for many destinations, there exists an alternate next-hop that will not lead to a forwarding loop. If such alternate paths exist for *all* traffic that is routed through a node, we can rely on LFA instead of protecting the node using MRC.

Based on the created configurations, we measure the backup path lengths (hop count) achieved by our scheme after a node failure. For a selected class of topologies, we evaluate how the backup path lengths depend on the number of backup Configurations.

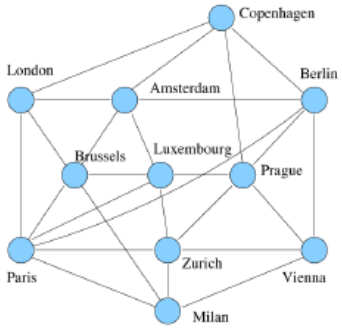

Fig. 4. The COST239 network.

The COST239 network is selected for this evaluation because of its resilient network topology. By using this network, we avoid a situation where there exists only one possible backup path to a node. The differences with respect to link loads between different recovery strategies will only be visible when there exists more than one possible backup path. In the COST239 network each node has a node degree of at least four, providing the necessary manoeuvring space. For our load evaluations, we use a gravity-style traffic matrix where the traffic between two destinations is based on the population of the countries they represent for simplicity; we look at constant packet streams between each node pair. The traffic matrix has been scaled so that the load on the most utilized link in the network is about 2/3 of the capacity. We use shortest path routing with equal splitting of traffic if there exists several equal cost paths towards a destination.

5.2 Number of Backup Configurations

Fig. 5 shows the minimum number of backup configurations that Algorithm 1 could produce in a wide range of synthetic topologies. Each bar in the figure represents 100 different topologies given by the type of generation model used, the links-to-node ratio, and the number of nodes in the topology. Table II shows the minimum number of configurations Algorithm 1 could produce for selected real-world topologies of varying size. For the Sprint US network, we show results for both the POP-level and router level topologies. The table also shows how many nodes that are covered by LFAs, and the number of configurations needed when MRC is used in combination with LFAs. Since some nodes and links are completely covered by LFAs, MRC needs to isolate fewer components, and hence the number of configurations decreases for some topologies.

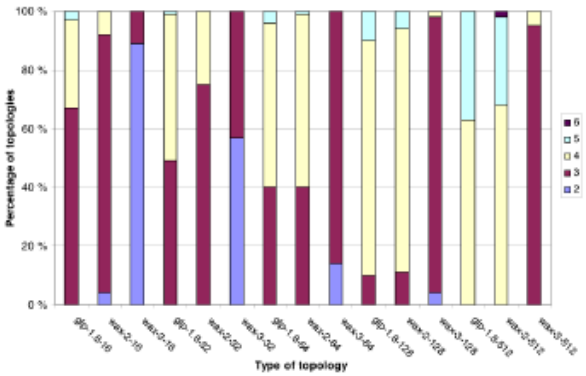

Fig. 5. The number of backup configurations required for a wide range of BRITE generated topologies. As an example the bar name wax-2-16 denotes that the Waxman model is used with a links-to-node ratio of 2, and with 16 nodes.

Table: 2

TABLE II
NUMBER OF BACKUP CONFIGURATIONS FOR SELECTED REAL-WORLD NETWORKS

Network	Nodes	Links	Confs	LFA	Confs
Sprint US (POP)	32	64	4	17	4
Sprint US (R)	284	1882	5	186	5
Geant	19	30	5	10	4
COST239	11	26	3	10	2
German Telecom	10	17	3	10	-
DFN	13	37	2	13	-

We see that for the COST239 network, all nodes except one is covered by LFAs. However, we still need two backup configurations to cover this single node, because isolating all the attached links in a single configuration would leave the node unreachable. The results show that the number of backup configurations needed is usually modest; 3 or 4 is typically enough to isolate every element in a topology. No topology required more than six configurations. In other words, Algorithm 1 performs very well even in large topologies.

5.3 Backup Path Lengths

Fig. 6 shows path length distribution of the recovery paths after a node failure. The numbers are based on 100 different synthetic Waxman topologies with 32 nodes and 64 links

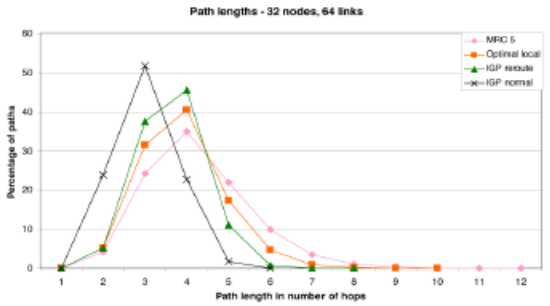

Fig. 6. Backup path lengths in the case of a node failure.

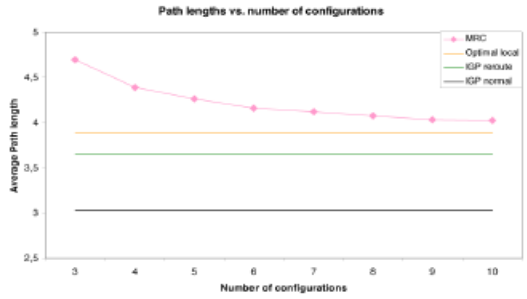

Fig. 7. Average backup path lengths in the case of a node failure as a function of the number of backup configurations.

All the topologies have unit weight links, in order to focus more on the topological characteristics than on a specific link weight configuration. Results for link failures show the same tendency and are not presented. For reference, we show the path length distribution in the failure-free case ("IGP normal"), for all paths with at least two hops. For each of these paths, we let every intermediate node fail, and measure the resulting recovery path lengths using global IGP rerouting, local rerouting based on the full topology except the failed component ("Optimal local"), as well as MRC with 5 backup configurations. We see that MRC gives backup path lengths close to those achieved after a full IGP re-convergence. This means that the affected traffic will not suffer from unacceptably long backup paths in the period when it is forwarded according to an MRC backup configuration.

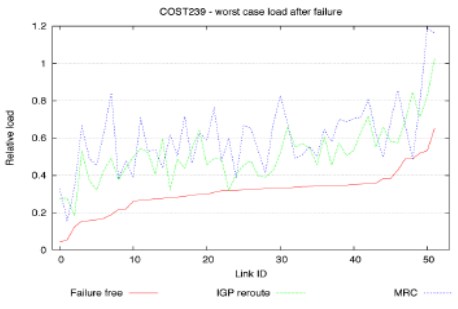

Fig. 8. Load on all unidirectional links in the failure free case, after IGP re-convergence, and when MRC is used to recover traffic. Shows each individual links worst case scenario.

Fig. 8 shows the maximum load on all links, which are indexed from the least loaded to the most loaded in the failure-free case. The results indicate that the restricted routing in the backup topologies result in a worst case load distribution that is comparable to what is achieved after a complete IGP rerouting process. However, we see that for some link failures, MRC gives somewhat higher maximum link utilization in this network. The maximum link load after the worst case link failure is 118% with MRC, compared to 103% after a full IGP re-convergence. In the next section, we discuss a method for improving the post failure load balancing with MRC.

6. RECOVERY LOAD DISTRIBUTION

MRC recovery is local, and the recovered traffic is routed in a backup configuration from the point of failure to the egress node. This shifting of traffic from the original path to a backup path affects the load distribution in the network, and might lead to congestion. In our experience, the effect a failure has on the load distribution when MRC is used is highly variable. Occasionally the load added on a link can be significant, as we saw in Fig. 8. In this section, we describe an approach for minimizing the impact of the MRC recovery process on the post failure load distribution Network operators often plan and configure their network based on an estimate of the traffic demands from each ingress node to each egress node.

Clearly, the knowledge of such demand matrix provides the opportunity to construct the backup configurations in a way that gives better load balancing and avoids congestion after a failure. We propose a procedure to do this by constructing a complete set of valid configurations in three phases. First, the link weights in the normal configuration are optimized for the given demand matrix while only taking the failure free situation into account. Second, we take advantage of the load distribution in the failure free case to construct the MRC backup configurations in an intelligent manner. Finally, we optimize the link weights in the backbones of the backup configurations to get a good load distribution after any link failure.

7. RELATED WORK

Much work has lately been done to improve robustness against component failures in IP networks in this section; we focus on the most important contributions aimed at restoring connectivity without a global re-convergence. We indicate whether each mechanism guarantees one-fault tolerance in an arbitrary bi-connected network, for link and node failures, independent of the root cause of failure (failure agnostic).
We also indicate whether they solve the "last hop problem". Network layer recovery in the timescale of milliseconds has traditionally only been available for networks using MPLS with its fast reroute extensions. In the discussion below, we focus mainly on solutions for connectionless destination-based IP routing.

8. CONCLUSION

We have presented Multiple Routing Configurations as an approach to achieve fast recovery in IP networks. MRC is based on providing the routers with additional routing configurations, allowing them to forward packets along routes that avoid a failed component. MRC guarantees recovery from any single node or link failure in an arbitrary bi-connected network. By calculating backup configurations in advance, and operating based on locally available information only, MRC can act promptly after failure discovery. MRC operates without knowing the root cause of failure.

9. REFERENCES

[1] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 2, pp. 35–44, Jul. 2005.

[2] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of failures in an IP backbone network," in *Proc. IEEE INFOCOM*, Mar. 2004, vol. 4, pp. 2307.

[3] S. Nelakuditi, S. Lee, Y. Yu, Z.-L. Zhang and C.-N. Chuah, "Fast local rerouting for handling transient link failures," *IEEE/ACM Trans. Networking*, vol. 15, no. 2, pp. 359–372, Apr. 2007. [9] S. Iyer, S. Bhattacharyya, N. Taft, and C. Diot, "An approach to alleviate link overload as observed on an IP backbone," in *Proc. IEEE INFOCOM*, Mar. 2003, pp. 406–416.

[4] S. Rai, B. Mukherjee, and O. Deshpande, "IP resilience within an autonomous system: Current approaches, challenges, and future directions," *IEEE Commun. Mag.*, vol. 43, no. 10, pp. 142–149, Oct. 2005.

[5] S. Bryant, M. Shand, and S. Previdi, "IP fast reroute using not-via addresses," Internet Draft (work in progress), draft-ietf-rtgwg-ipfrrnotvia- addresses-01, Jun. 2007.

[6] P. Francois, M. Shand, and O. Bonaventure, "Disruption free topology reconfiguration in OSPF networks," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 2007, pp. 89–97.