

IP Fast Reroute with Failure Inferencing

Junling Wang, Srihari Nelakuditi
 University of South Carolina, Columbia
 {wang257, srihari}@cse.sc.edu

Abstract—*Five nines availability is being expected from IP networks due to the growing popularity of IP telephony and the increasing usage of the Internet for mission-critical applications. This necessitates enhancing the resiliency of IP networks against transient failures that are observed to happen relatively frequently even in well-managed networks. Towards that end, we proposed failure inferencing based fast rerouting (FIFR) approach that exploits the existence of a forwarding table per line-card, for lookup efficiency in current routers, to provide fast rerouting similar to MPLS, while adhering to the destination-based forwarding paradigm. Earlier, we have shown that FIFR can deal with either single link or single node failures in a network consisting of point-to-point links with symmetric link weights. In this paper, we generalize FIFR to handle both link and node failures also in networks with asymmetric link weights and multi-access links. Furthermore, we extend FIFR to offer protection against the failure of an inter-AS link or an AS border node in case of a multi-homed AS. With these extensions, we argue that FIFR elevates the resiliency of any IP network with minimal changes to the forwarding and routing planes.*

I. INTRODUCTION

With the increasing dependence on the Internet for delivering various critical services, it is expected to be always available. In particular, applications such as Voice over IP demand *five-nines availability* (99.999% uptime) from IP networks as is the case with traditional telephone networks [1]. But it has been observed that transient failures happen relatively frequently even in well-managed IP backbone networks [2]. The existing IGP's such as OSPF and IS-IS typically take several seconds to converge and resume forwarding after a failure [3], whereas failure recovery within 50 ms is considered desirable for mission-critical applications [4]. Recent studies have shown that the IGP convergence can be accelerated to sub-second by tuning some parameters of these routing protocols [5], but bringing it down to sub-50ms level can potentially cause instability in the network, particularly due to *hot-potato routing* [6]. MPLS provides fast local rerouting effectively with its label stacking capability [7], but it is not scalable, as it requires a careful configuration of many backup label-switched paths for protection [8]. As an alternative, fast reroute mechanisms for pure IP networks have been proposed that make use of loop-free alternates [9], u-turn alternates [10], not-via addresses [11], or multiple routing configurations [12], for local rerouting upon a failure. While each of these mechanisms have their relative strengths, they either do not reroute to all destinations or rely on encapsulation or marking of datagrams for rerouting. We are interested in an IP fast reroute scheme that requires little or no changes to datagram format, forwarding process, or routing protocol while ensuring forwarding continuity to all reachable destinations.

We proposed a *failure inferencing based fast rerouting* (FIFR) approach earlier for handling transient failure of a link [13] or a node [14] in an IP network, leveraging the existence of a forwarding table per line-card of a router. Under FIFR, routers prepare for failures by computing *interface-specific forwarding* tables, i.e., a packet's next hop depends on both its destination address and incoming interface. A router, without being explicitly notified of a failure, can *infer* it from the incoming interface and destination address, and *precompute* interface-specific forwarding entries excluding the inferred failures. When a link/node fails, adjacent router suppresses global advertisement and instead initiates local rerouting of packets that were to be forwarded through the failed link/node. All other routers simply forward packets according to their precomputed interface-specific forwarding tables without relying on network-wide link-state advertisements. The attraction of FIFR is that the only change needed to the existing routers, with a line-card per interface, for providing protection against single link/node failures, is to replace the Dijkstra's algorithm for computing interface-independent forwarding entries with an algorithm to compute an additional interface-specific forwarding entry per destination. However, our earlier approach to handle node failures treats a link failure also as a failure of the corresponding tail node. This presumption could result in no route to the destination, even when there exists a path to it without the failed link, if the tail node partitions the network. Moreover, we assumed in our previous work on FIFR that the network consists of point-to-point links with symmetric link weights only.

We make several contributions in this paper to strengthen FIFR and make it applicable to a broad range of IP networks. First, we revise FIFR to guarantee loop-free forwarding to a destination if it is reachable, regardless of the failure of a link or a node in the network. Second, we generalize FIFR to handle failures in networks of bidirectional links with different link weights in each direction. This requires revamping of the procedures for inferring potential link/node failures and computing interface-specific forwarding entries. A noteworthy feature of the new version of FIFR is that it works fine in a network with a mix of symmetric and asymmetric link weights, and behaves exactly like the earlier version when all links have symmetric weights. Third, we show that FIFR is suitable even for networks with multi-access links by modelling each such link as a virtual node with an adjacency between it and each of the nodes adjacent to that link. Finally, we extend FIFR to offer protection against the failure of an AS peering link or an AS border node in case of a multi-homed AS. With these extensions, we believe FIFR would become a compelling alternative to the existing IP fast reroute schemes.

II. LINK AND NODE FAILURES

In this section, we introduce the FIFR approach for handling transient failures in a network consisting of point-to-point links with symmetric link weights. We present a more generalized FIFR suitable for networks with multiple access links and asymmetric link weights in later sections. Here, we first describe the general framework of the FIFR approach. We then present the details on how to handle link failures, followed by node failures, and finally both link and node failures.

A. FIFR Framework

Under FIFR, a router usually forwards a packet to the next-hop along the shortest path to its destination. But in case of a failure, the router adjacent to it locally reroutes packets that were affected by the failure to alternate next-hops without notifying all other routers about the failure. The central idea behind the FIFR approach is the ability of a router to infer potential non-adjacent failures from the incoming interface and the destination of a packet without being explicitly informed of the failure. When a packet arrives at a router through an unusual interface along the reverse shortest path, due to local rerouting by the router adjacent to the failure, through which it would never arrive had there been no failure, that router can identify the corresponding set of potential individual failures. Since these inferences can be made in advance, the forwarding entry associated with that interface and destination can be precomputed excluding the corresponding potential failures from the network topology. Due to such failure inferencing and interface-specific forwarding, in case of a single failure, FIFR can achieve both loop-freedom and destination-reachability, the two fundamental objectives of any fast rerouting scheme.

The packet forwarding under FIFR can be summarized as follows. Each router i under FIFR maintains a *routing table* entry \mathcal{R}_i^d per each destination d . In addition, it keeps a *forwarding table* entry $\mathcal{F}_{j \rightarrow i}^d$ and a *backwarding table* entry $\mathcal{B}_{i \rightarrow j}^d$ per each neighbor j and destination d . A packet originating at i to destination d is forwarded to \mathcal{R}_i^d . A packet destined for d arriving at i through neighbor j is forwarded to $\mathcal{F}_{j \rightarrow i}^d$. When an adjacent link $i \rightarrow j$ fails or adjacent node j fails, router i locally reroutes a packet destined for d to $\mathcal{B}_{i \rightarrow j}^d$, if it were to be forwarded to j had there been no failure.

The computation of \mathcal{R}_i^d and $\mathcal{B}_{i \rightarrow j}^d$ is rather straightforward whereas the computation of $\mathcal{F}_{j \rightarrow i}^d$ is more intricate but captures the essence of FIFR approach. Based on the notation in Table I,

$$\mathcal{R}_i^d = N(d, \text{SPT}(i, \mathcal{V}, \mathcal{E}))$$

The computation of $\mathcal{B}_{i \rightarrow j}^d$ depends on whether the adjacent failure is treated as a link or a node failure. Assuming we are handling only link failures, for ease of illustration, we have

$$\mathcal{B}_{i \rightarrow j}^d = N(d, \text{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{i \rightarrow j\}))$$

Again, assuming only link failures, before computing $\mathcal{F}_{j \rightarrow i}^d$, we need to infer $\mathcal{K}\mathcal{L}_{j \rightarrow i}^d$, a *key link* failure among all the potential failures that would cause a packet to destination d arrive at node i through neighbor j . We can then compute $\mathcal{F}_{j \rightarrow i}^d$ by removing $\mathcal{K}\mathcal{L}_{j \rightarrow i}^d$ from the network topology as follows.

$$\mathcal{F}_{j \rightarrow i}^d = N(d, \text{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{\mathcal{K}\mathcal{L}_{j \rightarrow i}^d\}))$$

TABLE I
NOTATION

\mathcal{V}	set of all vertices
\mathcal{E}	set of all edges
$E(n)$	set of all edges adjacent to node n
\mathcal{R}_i^d	set of next hops from i to d
$\mathcal{F}_{j \rightarrow i}^d$	set of next hops from $j \rightarrow i$ to d .
$\mathcal{B}_{i \rightarrow j}^d$	set of back hops from $i \rightarrow j$ to d .
$\mathcal{K}\mathcal{L}_{j \rightarrow i}^d$	the key link corresponding to $j \rightarrow i$ and d .
$\mathcal{K}\mathcal{N}_{j \rightarrow i}^d$	the key node corresponding to $j \rightarrow i$ and d .
$\text{SPT}(i, \mathcal{V}', \mathcal{E}')$	shortest path tree rooted at i w.r.t. the graph $(\mathcal{V}', \mathcal{E}')$
$\text{rSPT}(i, \mathcal{V}', \mathcal{E}')$	reverse SPT rooted at i w.r.t the graph $(\mathcal{V}', \mathcal{E}')$
$N(d, \mathcal{T})$	next hops to d from root of SPT \mathcal{T}
$P(d, \mathcal{T})$	previous hops from d to root of rSPT \mathcal{T}

B. Identifying Key Links

We now describe how a key link is identified for each interface and destination. A link $u \rightarrow v$ is considered to be a *candidate* key link w.r.t. interface $j \rightarrow i$ and destination d , if it satisfies both of the following conditions:

- 1) *with* $u \rightarrow v$, j is a next hop from i to d .
- 2) *without* $u \rightarrow v$, shortest path from u to d contains $j \rightarrow i$.

In other words, $u \rightarrow v$ is a candidate key link if it is along the shortest path from i via j to d , and its failure causes a packet for d to arrive at i through j along the *reverse* shortest path. Since these candidate links are common to all the shortest paths from i to d [13], we can find the one closest to the destination, referred to as the key link $\mathcal{K}\mathcal{L}_{j \rightarrow i}^d$. Once the key link is identified, the forwarding entry can be computed as shown earlier by excluding it from the network topology. We have shown that if there exists a path from i to d without the failed link, there exists a path without the key link [13].

Consider the topology shown in Fig. 1 where each link is labelled with its weight. When a packet destined to F arrives at A via B, router A can infer that either B-E or E-F must have been down. Therefore, the key link corresponding to interface B-A and destination F, $\mathcal{K}\mathcal{L}_{B \rightarrow A}^F$ is E-F since it is closer to F than B-E. Consequently, $\mathcal{F}_{B \rightarrow A}^F$ is H, i.e., A would forward the packet to H ensuring that it reaches destination F without actually knowing whether B-E or E-F failed.

C. Identifying Key Nodes

Analogous to key link inferencing, we can also identify key nodes assuming only node failures. A node v is considered to be a candidate key node w.r.t. interface $j \rightarrow i$ and destination d , if it satisfies both of the following conditions:

- 1) *with* v , j is a next hop from i to d .
- 2) *without* v , the shortest path from a parent node of v (w.r.t. $\text{SPT}(i, \mathcal{V}, \mathcal{E})$) to d contains $j \rightarrow i$.

Among all the candidates, the one closest to d is referred to as the key node, $\mathcal{K}\mathcal{N}_{j \rightarrow i}^d$. Hence, for handling node failures, the forwarding and backwarding table entries would be

$$\mathcal{B}_{i \rightarrow j}^d = N(d, \text{SPT}(i, \mathcal{V} \setminus \{j\}, \mathcal{E} \setminus E(j)))$$

$$\mathcal{F}_{j \rightarrow i}^d = N(d, \text{SPT}(i, \mathcal{V} \setminus \{\mathcal{K}\mathcal{N}_{j \rightarrow i}^d\}, \mathcal{E} \setminus E(\mathcal{K}\mathcal{N}_{j \rightarrow i}^d)))$$

The shortest path from node s to node d in an asymmetric network is not the exact reverse of that from d to s . For example, in Fig. 2, node E reaches B directly while B reaches E through A. This leads to the forwarding loop problem in asymmetric networks since link weights used during usual forwarding and local rerouting upon failures are inconsistent. To enforce consistency, upon a failure adjacent to s , we require that a packet from s to d is forwarded along $\text{rrSP}(s, d)$, i.e., the reverse path of the shortest path from d to s . In order to compute rrSP , a router can build the reverse shortest path tree rSPT , where each path from the root represents a rrSP . Similarly, non-adjacent routers infer failures and compute forwarding entries for unusual interfaces using rSPT as described below. Note that rrSP is only used for local rerouting upon a failure to affected destinations, while the conventional shortest path is still used for failure-free forwarding.

Let us consider the previous example again where a packet is being forwarded from C to D when C–D is down. The shortest path from D to C without C–D is $D \rightarrow G \rightarrow F \rightarrow E \rightarrow B \rightarrow C$. Therefore, $\text{rrSP}(C, D)$ excluding C–D is $C \rightarrow B \rightarrow E \rightarrow F \rightarrow G \rightarrow D$. Node B, which is not adjacent to the failed link C–D, can infer the failure associated with the corresponding reverse usual interface by computing the rrSP to D, and forward it to E instead of A as before. Node E would also infer similarly as B and forward the packet to F. Since E is not the usual next hop of F to D, F would forward the packet to its usual next-hop which is node D itself. Thus, a packet from C arrives at D along a loop-free path $C \rightarrow B \rightarrow E \rightarrow F \rightarrow D$.

The computation of backwarding and forwarding table entries have to be done differently to accommodate asymmetric link weights. First, backwarding entries would be,

$$\mathcal{B}_{i \rightarrow j}^d = P(d, \text{rSPT}(i, \mathcal{V} \setminus \{j\}, \mathcal{E} \setminus E(j)))$$

If $\mathcal{B}_{i \rightarrow j}^d$ is \emptyset , then

$$\mathcal{B}_{i \rightarrow j}^d = P(d, \text{rSPT}(i, \mathcal{V}, \mathcal{E} \setminus \{i-j\}))$$

The computation of forwarding entries involves redefining the conditions for identifying key link and key node candidates.

A link $u \rightarrow v$ is a candidate key link w.r.t. $j \rightarrow i$ and d , if

- 1) with $u \rightarrow v$, j is a next hop from i to d .
- 2) without $u \rightarrow v$, $\text{rrSP}(u, d)$ contains $j \rightarrow i$.

A node v is a candidate key node w.r.t. $j \rightarrow i$ and d , if

- 1) with v , j is a next hop from i to d .
- 2) without v , the rrSP from a parent node of v (w.r.t. $\text{SPT}(i, \mathcal{V}, \mathcal{E})$) to d contains $j \rightarrow i$.

The key node $\mathcal{KN}_{j \rightarrow i}^d$ and key link $\mathcal{KL}_{j \rightarrow i}^d$ will still be those closest to d among the corresponding candidates. Now, if $\mathcal{KN}_{j \rightarrow i}^d \neq \emptyset$,

$$\mathcal{F}_{j \rightarrow i}^d = P(d, \text{rSPT}(i, \mathcal{V} \setminus \{\mathcal{KN}_{j \rightarrow i}^d\}, \mathcal{E} \setminus E(\mathcal{KN}_{j \rightarrow i}^d)))$$

else if $\mathcal{KL}_{j \rightarrow i}^d \neq \emptyset$,

$$\mathcal{F}_{j \rightarrow i}^d = P(d, \text{rSPT}(i, \mathcal{V}, \mathcal{E} \setminus \{\mathcal{KL}_{j \rightarrow i}^d\}))$$

otherwise,

$$\mathcal{F}_{j \rightarrow i}^d = N(d, \text{SPT}(i, \mathcal{V}, \mathcal{E}))$$

In a network with symmetric links only, the above formulation yields the same backwarding and forwarding entries as before.

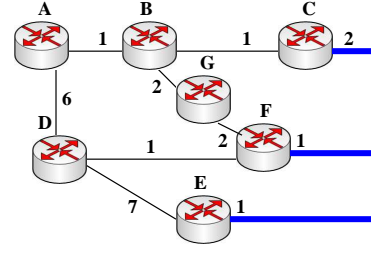


Fig. 3. A network with routers on a LAN

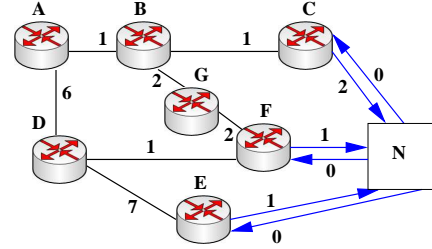


Fig. 4. A representation of the LAN

IV. MULTI-ACCESS LINKS

The description of FIFR so far assumed that the network consists of only point-to-point links such that each router knows the neighbor attached to the incoming interface of a packet, enabling it to infer failures. It is not obvious whether FIFR can be generalized to work with broadcast LANs and non-broadcast multi-access (NBMA) links where multiple neighbors are attached to a router through the same interface. It appears as if FIFR can not be employed in such networks, since a router seems not to know which neighbor is forwarding it the packet. However, we show that it is still possible to infer non-adjacent failures even in the presence of multi-access links. In particular, by utilizing the topology representation defined in IGP (particularly in OSPF [16]), FIFR can be applied almost as is to work with multi-access links. In the following, we discuss several issues pertaining to the application of FIFR to networks with multi-access links.

A. Failure Detection

Compared to point-to-point links where a failure is either a link or node failure, broadcast links have more possible scenarios of failures. We consider, w.r.t. a node i and a next-hop node j , the failure of one of the following: 1) interface of i to LAN; 2) LAN itself 3) interface of j to LAN 4) node j . We treat the first two as LAN failure and the last two as node failure. We assume that the linecard hardware of a router attached to a broadcast LAN can detect the failure of its connection to LAN, as in SDH/SONET and Gigabit Ethernet, which could signal either the LAN failure or the interface failure. In order to handle the failure promptly, we consider it as the LAN failure without further diagnosis of the failure type. However, in general there is no easy way for a router to learn from the hardware about the failure of another router on the broadcast LAN. Bidirectional Forwarding Detection (BFD [17]), an implementation of faster hello, can be used in this case to detect the individual router failure.

B. Modeling Broadcast Links

Instead of modeling each pair of routers that can directly communicate in a broadcast LAN as a point-to-point link, OSPF elects a Designated Router (DR) for the LAN. Only the DR forms the adjacency to the other routers in the LAN and is responsible for generating LSAs for the LAN itself. In OSPF, a broadcast LAN is represented as a virtual node in the link state database. This reduces the number of adjacencies required in the network and thus reduces the protocol messages and link state database size. More importantly, it fits FIFR so well that we can apply FIFR almost as is to the topology where a broadcast LAN is modeled as a virtual node.

Fig. 3 illustrates a simple example of a routing domain that contains a LAN connecting three routers, C, F and E. The LAN is represented as a node N in the link state database as shown in Fig. 4. Note that the cost of edges from the LAN to routers are always 0. By representing the LAN as a virtual node in the topology graph, the LAN interfaces are modeled as point-to-point links between the routers and the virtual LAN node and the topology becomes a graph with asymmetric links. Thus, FIFR described in the previous section can be applied without any hitches. The following details how non-adjacent nodes infer failures in a routing domain with broadcast LANs.

1) *Inference of LAN Failure:* As explained earlier, an adjacent router treats the failure associated with a LAN as if the LAN itself failed. The other non-adjacent routers can infer the LAN failure using the usual key node definition described in the previous section. Since the LAN is modeled as a virtual node, FIFR does not infer the link failure for all links between the routers and the LAN node. For example, in Fig. 4, links C-N, F-N and E-N are not considered to fail separately and therefore they can not be the key links for any interface.

2) *Inference of LAN Router Failure:* When a router does not receive a BFD response from another LAN router, it treats the failure as a node failure if its failure does not partition the routing domain. For example, without any failure, C would forward a packet destined for D to F over the LAN. When C does not receive BFD response from F, it simply considers it as the failure of node F. Node C then computes the alternate rSP, which would be $C \rightarrow B \rightarrow A \rightarrow D$ and forwards the packet to B. Again, the other routers can infer the node failure through the usual definition of the key node whose failure will cause its parent node to reroute the packet through the reverse usual interface. We want to stress that when FIFR is applied to LANs, the parent node of a router on LAN can not be the virtual LAN node as suggested by the topology, since the LAN node itself can not make rerouting decisions. In the previous example, the shortest path from B to D is $B \rightarrow C \rightarrow N \rightarrow F \rightarrow D$. However, when B infers the key node, C should be considered the parent of node F instead of N. Otherwise, if N is treated as the parent node, the failure of F causes N to reroute the packet through E to D, not through $C \rightarrow B$. Thus, F would not be the key node for interface $C \rightarrow B$ and destination D. So, when B receives a packet from C, it infers only the LAN failure and forwards the packet to G, causing the packet to oscillate between G and C. Except for taking care of this, FIFR can be applied as is to networks with multi-access links.

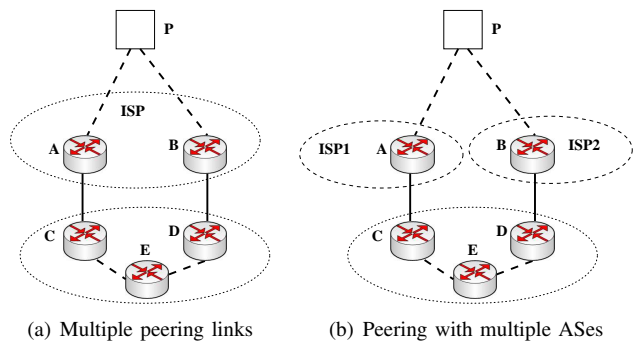


Fig. 5. AS with multiple egress nodes to a destination prefix

When the failure of a router on a LAN partitions the routing domain or specifically the router itself is the destination, it is not appropriate to treat it as the node failure. In this case, the router adjacent to the failure attempts to deliver the packet to its destination by encapsulating the packet with the same destination and forwards it along an alternate path after excluding the virtual LAN node. This is similar to the procedure we discussed in the previous section on merging key links and nodes. For example, if C has a packet destined for F, when it does not get BFD response from F, C finds an alternate path through B to F after excluding the LAN. In case node F indeed failed, the other adjacent node G will discard the encapsulated packet as it indicates that the packet has already encountered a failure before. In summary, the generalized version of FIFR protects against both link and node failures in an IP network with point-to-point/multi-access links with symmetric/asymmetric weights.

V. INTER-AS FAILURES

The discussion of FIFR thus far has been about dealing with failures inside a routing domain, i.e., an OSPF area within an Autonomous System (AS). We now show that with proper interaction between IGP and BGP as defined in [18], FIFR can be extended to handle not only intra-AS failures but also inter-AS failures. Though FIFR can be extended to work with multiple areas in an AS but for simplicity, in this section, we assume that an AS consists of a single area.

The same problem of BGP peering link failures is addressed in [19] by relying on provisioning of protection tunnels from the primary egress to the secondary egress. In contrast, FIFR does not require setting up of additional protection tunnels. By importing BGP routes into IGP, a router can obtain information on the external routes and the interface specific forwarding entry computed by FIFR can automatically protect against both intra-AS or inter-AS failures. FIFR is applicable to both large transit AS and multihomed AS. An example of an AS with two egress nodes to some common prefixes is illustrated in Fig. 5. Those egress nodes can connect to one ISP (5(a)) or two ISPs (5(b)). Assume C is the primary egress node. In either of the two cases, FIFR can automatically switch the path from primary to secondary egress node if inter-AS link fails.

Node C and D are generally the Autonomous System Border Routers (ASBR) that run both BGP and IGP protocols. From eBGP [20], they learn routes to the prefix P via A

and B respectively that is specified in the BGP next hop attribute. After ASBR imports BGP routes into OSPF, these next hops are advertised through the “forwarding address” field in LSA to internal OSPF routers. According to the standard of BGP4/IDRP for IP–OSPF interaction [18], the value carried by the LOCAL_PREF attribute in BGP has no impact on the route selection and egress node is determined solely by the OSPF cost. That requires the administrators to specify the cost for these external learned routes. Once the costs for these routes are specified, each internal router has a view of topology as illustrated in Fig. 5. Therefore, FIFR can be applied as is to the topology incorporated with external routes. We discuss the behavior of FIFR behavior when it is applied in such a setting.

- 1) FIFR will reroute packets to the backup egress node automatically in case that inter-AS link between primary egress node and ISP is broken. In general, A is the key node to the destination P and its failure can be inferred by internal routers and therefore the packet can be consistently forwarded to the secondary egress B.
- 2) Occasionally, FIFR may reroute packets to the backup provider on few intra-AS failures even if the primary inter-AS link is still functioning. It happens because failure inference in FIFR is inclusive and intra-AS failures cause some routers to infer the inter-AS link failure. It may seem strange that FIFR switches to the backup provider when an intra-AS failure occurs and the path to the primary provider still exists. But it only lasts for a short period of time after the failure while it is suppressed and the traffic will switch back to the primary provider after the global convergence.
- 3) In case that only one provider exists, the key node then can not be the next hop of the egress point, which means that local failures within an AS do not cause FIFR to infer inter-AS failures. For example, assume that ISP2 does not exist in Fig. 5(b). In this case, node A can not be the key node to P for any interface, since its failure would not cause the packet to be rerouted but to be discarded. Therefore, on all intra-AS failures within a stub AS, FIFR would attempt to find an alternate route to its only egress node.

VI. CONCLUSIONS

The availability of IP networks can be enhanced when transient failures are handled locally and local rerouting is initiated immediately on detection of these failures. In this paper, we described a *failure inferencing based fast rerouting* approach for local rerouting around failed links and nodes without explicit link state updates. Our approach is based on failure inferencing associated with reverse usual interfaces. We revised our earlier work by merging inference of key links and nodes to guarantee loop-free forwarding whenever the destination is reachable. FIFR in general assumes node failures but considers link failures when node failures cause the destination not reachable. Packet encapsulation is performed by the router adjacent to a failure when the router treats it as a link failure. With one-level encapsulation requirement, FIFR can avoid forwarding loops when a node indeed fails.

We further generalized FIFR to handle networks with asymmetric link weights also by finding alternate paths using reverse shortest path trees, which yield the usual shortest paths if the network contains symmetric links only. Therefore, the generalized version of FIFR need not distinguish between the symmetric and asymmetric networks and is designed to work for both. By representing the multi-access links as a virtual node, FIFR can deal with them also with slight modification only. Furthermore, if IGP and BGP are configured to interact properly at the ASBRs, FIFR can be extended to protect against both intra-AS and inter-AS failures.

REFERENCES

- [1] Nokia Communications, “The Five Nines IP Network,” White Paper, Jan. 2002.
- [2] A. Markopulu, G. Iannaccone, S. Bhattacharya, C.-N. Chuah, and C. Diot, “Characterization of failures in an IP backbone,” in *Proc. IEEE Infocom*, Mar. 2004.
- [3] Gianluca Iannaccone, Chen-Nee Chuah, Richard Mortier, Supratik Bhattacharyya, and Christophe Diot, “Analysis of link failures in an IP backbone,” in *Proc. ACM Sigcomm Internet Measurement Workshop*, Nov. 2002.
- [4] J.-P. Vasseur, M. Pickavet, and P. Demeester, *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*, Morgan Kaufmann, 2004.
- [5] Pierre Francois, Clarence Filsfils, John Evans, and Olivier Bonaventure, “Achieving sub-second igp convergence in large ip networks,” *ACM SIGCOMM Computer Communications Review*, vol. 35, no. 2, pp. 35–44, July 2005.
- [6] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford, “Dynamics of hot-potato routing in IP networks,” in *Proc. ACM Sigmetrics*, June 2004.
- [7] V. Sharma et al., “Framework for MPLS-based recovery,” IETF Internet Draft, Jan. 2002, draft-ietf-mpls-recovery-frmrk-03.txt.
- [8] Avici Systems, “Reliable Alternate Paths for IP Destinations,” White Paper, Apr. 2004.
- [9] A. Atlas, R. Torvi, G. Choudhury, C. Martin, and B. Imhoff, “Loop-Free Alternates for IP/LDP Local Protection,” Internet Draft(work in progress), May 2005, draft-atlas-ip-local-protect-loopfree-00.txt.
- [10] A. Atlas, “U-turn alternates for IP/LDP fast-reroute,” IETF Internet Draft, Feb. 2005, draft-atlas-ip-local-protect-uturn-02.txt.
- [11] S. Bryant, M. Shand, and S. Previdi, “IP fast reroute using not-via addresses,” Internet Draft(work in progress), Mar. 2006, draft-bryantshand-IPFRR-notvia-addresses-02.txt.
- [12] A. Kvalbein, A.F. Hansen, T. Cicic, S. Gjessing, and O. Lysne, “Fast IP Network Recovery using Multiple Routing Configurations,” in *Proc. IEEE Infocom*, Apr. 2006.
- [13] 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.
- [14] Z. Zhong, S. Nelakuditi, Y. Yu, S. Lee, J. Wang, and C.N. Chuah, “Failure inferencing based fast rerouting for handling transient link and node failures,” in *GI*, Mar. 2005.
- [15] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, “Generic Routing Encapsulation (GRE),” RFC 2784, Mar. 2000.
- [16] J. Moy, “OSPF Version 2,” RFC 2328, Apr. 1998.
- [17] D. Katz and D. Ward, “Bidirectional Forwarding Detection,” draft-ietf-bfd-base-06.txt, Mar. 2007.
- [18] K. Varadhan, S. Hares, and Y. Rekhter, “BGP4/IDRP for IP–OSPF Interaction,” RFC 1745, Dec. 1994.
- [19] O. Bonaventure, C. Filsfils, and P. Francois, “Achieving Sub-50 Milliseconds Recovery Upon BGP Peering Link Failures,” Oct. 2005.
- [20] Y. Rekhter and T. Li, “A Border Gateway Protocol 4 (BGP-4),” RFC 1771, Mar. 1995.