Could Proactive Link-State Routed Wireless Networks Benefit from Local Fast Reroute?

Audun Fosselie Hansen[1,2], Geir Egeland[2] and Paal Engelstad[2]
[1] Networks and Distributed Systems Group, Simula Research Laboratory, Fornebu, Norway
{audunh}@simula.no
[2]Telenor R&I, Fornebu, Norway, {geir.egeland, paal.engelstad}@telenor.com

Abstract

The communication performance in wireless networks is often heavily influenced by failures caused by node mobility and radio disturbance. Proactive link-state routing protocols like OLSR and OSPFv3 with wireless extensions (WOSPF) are currently relying on slow re-convergence to restore successful packet forwarding. This may not be appropriate for particular applications, and it may cause instability during transient failures. Support for fast and local reroute should therefore be added as a feature to proactive link-state based routing protocols. In this paper, we discuss the challenges and requirements that are associated with such solutions. In addition, we evaluate and discuss to what extent recent developments for fixed IP routing can serve as solutions.

1 Introduction

Standards for wireless communication technologies are continuously being proposed, and existing ones are evolving with more features. All kinds of applications are envisioned to be offered in these networks. Some might have stringent requirements to robust delivery of data, either due to a real-time nature of the application or due to the importance of the content. These applications bring challenges to the routing protocols due to the failure frequency of wireless links and possible mobility of nodes. A considerable amount of these failures are transient, meaning that the links and nodes may return to normal operation and position within short time. In such scenarios, routing protocol actions like re-convergence of routing state may cause instability and unnecessary bandwidth consumption.

A typical solution to provide fast recovery for critical traffic and to prevent instability during transient failures is to pre-establish multiple paths between the source an destination. Then an alternative path can be used during transient failure periods. Reactive routing protocols like AODV and DSR have been extended to support multipath routing [12–15, 17, 18, 22]. The most common multipath approach is to establish disjoint paths from the source. Loop-free forwarding is ensured by having a path identifier like in connection-oriented routing. It is not always possible for the node that detects a failure to perform the rerouting action, and it must therefore send a failure notification to the source or a neighbor node that knows about an alternative path. This may impose some delay on the recovery action.

In this paper, we will focus on proactive link-state routing protocols. More specifically, we will address routing protocols that are connectionless and where routing decisions are taken hop-by-hop based on the destination address in the packet header. In the EU IST HIDENETS[1] project, we have identified several application scenarios (e.g. platooning and brigade communication) that could benefit from fast reroute support in such proactive routing protocols [19].

Typical routing protocols in this category are WOSPF [3,9], OLSR [6] and TBRPF [16]. Since, there is no per source-destination route discovering process, building multiple paths has not been an option in such networks. In addition, they operate in a pure connectionless manner, and path identification is not supported. Currently, solving local fast reroute is a very hot topic in connectionless fixed IP networks. The basic idea is that each node should, based on the link state information, store a backup next hop to be used if the primary next hop fails. A node that detects a failure towards its primary next hop should then locally start to send the affected traffic on a backup next hop. A solution to this is not straightforward due to challenges with for instance looping.

The next section will elaborate on requirements

[1]www.hidenets.aau.dk

and challenges with regards to such local fast reroute schemes. Recently, a set of schemes has been proposed to solve these challenges for fixed connectionless IP networks. In section 3, we present these schemes. We evaluate the performance in some wireless network scenarios. Section 4 presents the evaluation method and present the results. In section 4.3, we discuss and summarize the findings. Finally, section 5 concludes the paper.

2 Expectations to IP fast reroute

In this section, we present the basic requirements for a fast reroute scheme. In networks based on link-state routing and destination-based hop-by-hop forwarding, a solution to these requirements are not straightforward. We therefore describe associated challenges. A local fast reroute scheme may affect various performance metrics, and we will therefore present some of the most important performance metrics that should be considered when designing and evaluating a local fast reroute solution. In section 4, we will evaluate IP fast reroute schemes with respect to the following requirements, challenges and performance metrics.

2.1 Functional requirements

Local A fast reroute scheme must be local in order to offer really fast reroute. This means that the node detecting a failure initiates the rerouting process. A global scheme requires that at least the source nodes of the traffic are notified about the failure(s), a process that may be to time-consuming for recovering particular types of traffic.

Strict local rerouting can be an advantage of proactive link-state routed schemes over reactive multipath schemes where strictly local rerouting cannot always be supported.

Proactive A fast reroute scheme must be proactive in the sense that no route calculations or signaling are to be performed upon detection of a failure. All rerouting information must be configured and installed in advance. A reactive approach will cause a longer delay and potentially also more instability. Multipath approaches may improve the delay of a reactive reroute scheme, but still some signaling and failure notifications may be necessary upon detection of a failure.

Root cause of failure If a fast reroute scheme is supposed to protect both nodes and links, it must be able to route traffic correctly without knowing the root cause of failure, e.g. knowing whether it is a link or node that has failed. This requirement is necessary because most networks do not offer mechanisms that enable a node to quickly diagnose a failure towards the next hop.

2.2 Challenges to solve

Loop-free sequences of next hops The most intuitive way of doing IP fast reroute in proactive link-state routed networks would be to let a router forward packets to a backup next hop upon detection of a failure on the primary next hop. The challenges associated with such an approach are twofold. First, since only the detecting nodes are aware of the failure, other nodes that are not explicitly aware of the failure must be told how to treat packets. Second, the sequences of next hops must be coordinated in a way so that packets do not loop. This is not straightforward in connectionless IP networks where packets are routed hop-by-hop without any path identifier.

Last hop problem If a fast reroute scheme is supposed to handle both node and link failures without knowing the root cause of failure, a challenge denoted as the "last hop problem" must be considered. Normally, a scheme that is designed to handle node failure recovery implicitly addresses link failures too, as the adjacent links of the failed node can be avoided. This is true for intermediate nodes, but in the last hop case such a scheme will drop packets unnecessarily if it is only a link that has failed and not the entire node. Therefore, a fast reroute scheme must be designed so that only the link is protected in the last hop case.

Loop avoidance during not recoverable failures In a network that does not support fast reroute, nodes will drop packets if the normal next hop is not operational. If, however, a rerouting scheme is deployed, the nodes will attempt to forward packets to an alternative next hop. This gives planned behavior as long as there are not more failures in the network than the scheme can handle. Special care must therefore be taken to ensure that packets do not loop when there are more failures than the rerouting scheme can handle.

2.3 Performance metrics to consider

Recovery success Guaranteeing single link failure recovery, or preferably both single node and link failure recovery is a common requirement for recovery schemes. Furthermore, high recovery success when there is more failures will be regarded as an advantage. This is particularly important in wireless ad hoc network which are prone to node mobility and link interruptions.

Path lengths Path lengths or hop count is a very important metric for most ad hoc network technologies that exist today [7]. This is due to the broadcast nature of wireless communication. An extra hop does not only affect the two neighbors communicating, but all nodes that can hear the broadcasting. [4] shows that the throughput in the network decreases close to exponentially with the number of hops in an 802.11 network. We must add that the penalty of extra hops may be less severe in other and future technologies; however higher penalty than in wired switched networks will always be the case.

State requirements Dependent on the capacity of the nodes, the amount of state imposed by the fast reroute scheme may be an important metric. Both storage capacity and computation capacity to calculate the state will determine the importance of state requirements.

Computation Complexity For a proactive recovery scheme, the backup routes must be calculated in advance. The importance of simple calculation will be dependent on the computation capacity of the nodes.

Forwarding complexity With forwarding complexity we mean the complexity of the actions taken by a node from it receives a packet to the packet is delivered to an appropriate neighbor. This is also most important when the nodes has limited computation capacity.

Flexibility for optimizations Different types of networks and scenarios might require different types of optimizations. A fast reroute scheme that provides flexibility along several optimization axis could be considered stronger than schemes that do not offer this flexibility. Flexibility for traffic engineering and QoS optimization for the rerouted traffic could be one feature. Easy support for handling multiple failure can be another feature.

3 Schemes to be evaluated

In this section, we present the different fast reroute schemes that will be evaluated in this paper. We implement all the schemes so that they are optimized for handling link failures and both node and link failures. The different optimizations will be evaluated in section 4. We compare the performance with the "Normal" case where no rerouting exist. In addition, we compare with simple "Deflection", which deflects packets to a random next hop upon detection of a failure.

All schemes except deflection use full topology information to build backup next hops that can be used upon detection of failures. This information is retrieved from link state dissemination using a proactive link state protocol. Not-via, FIFR, IPRT and DMRC guarantee 100 % recovery from any single failure in biconnected topologies. Deflection and loop free alternates (LFA) do not provide any recovery guarantees. All schemes meet the functional requirements listed in the previous section, and hence provide local and proactive recovery independent of root cause of failures.

Normal

Normal denotes the normal shortest path routing that has been calculated on the basic original topology. With Normal routing, packets will be dropped if a node detects a failure in the next hop for the packet. This will be counted as an unsuccessful sample when measuring recovery success rates.

Deflection

With deflection, a node that detects a failure is allowed to deflect the packets to another random neighbor. If the node does not have another neighbor, the packets will be dropped. If, however, the nodes in a network have alternative neighbors, the packets will be forwarded in the network until they reach the destination, or are being discarded due to for instance reaching the TTL limit. Hence, there is a possibility that packets may loop in the network for a while. We count an unsuccessful sample when a TTL loop is discovered.

Deflection is not relying on pre-planning backup next hops, and hence loop-free sequences of backup next hops are not supported. Deflection is not affected by the last hop problem since recovery is not planned for either link or node failures. Deflection cannot prevent loops during not recoverable failures.

Loop-free alternates (LFA) [1]

Seen from a node, a loop-free backup next hop exist if the path via this next hop does not loop back to the failure. [1] presents some cost conditions that define valid backup next hops for protecting a neighbor link or neighbor node for a given destination. There is no guarantee that there exist such alternates for all links and nodes for all destinations in biconnected networks.

If the node that detects a failure has a loop-free alternative next hop, it forwards the packet to that alternative next hop. If no such next hop exist, the packet is dropped. An unsuccessful sample is counted when packets are dropped.

LFA recovery provides implicitly loop-free sequences of backup next hops. It sort of solves the last hop problem in that it can use only link LFA in the last hop. However, it cannot prevent loops when there are more failures than planned for. With multiple failures, packets may loop between nodes that try to use LFAs.

Tunneling using Not-via addresses (Not-via) [21]

Tunneling using Not-via addresses adapts the idea behind MPLS local detours to connectionless IP. The main idea is to route the packets around the failed component to the node after the failure in the forwarding path. This is done by addressing a packet to the node downstream of the failed component. Packets are dropped when a failure is detected while the packets are already in a Not-via tunnel. We then count an unsuccessful sample.

All nodes calculate Not-via paths based on the same topology view, and hence it provide loop-free sequences of backup next hops. It solves the last problem by building Not-via paths only for links in the last hop. It prevents loops when there are more than one failure by dropping packets that are already in a Not-via tunnel.

Failure Inferencing based Fast Rerouting (FIFR) [25]

FIFR uses neighbor specific forwarding. This means that a node decides the next hop for the packets based on both destination address and what neighbor the packets are coming from . FIFR does not require any packet marking or tunneling. However, the default Dijkstra shortest path tree calculation is replaced by new algorithms. FIFR offers no obvious way to infer that a packet has been rerouted, and hence packets may not be dropped when detecting more failures in the network. This may lead to packets looping in the network. We count an unsuccessful sample when such a loop occurs.

FIFR provides loop-free sequences of backup next hops due to the neighbor specific routing. FIFR can solve the last hop problem if only link protection is considered towards the destination. FIFR cannot prevent loops when there are to many failures in the network. This is because FIFR does not rely on packet marking and nodes detecting a second failure do not drop packets.

IP Redundant Trees (IPRT) [5, 20]

IPRT is based on building two trees rooted in and directed towards each destination. From these two trees it is possible to find backup next hops that can be used when a failure is detected. A node that detects a failure will then start forwarding packets according to such backup next hops. For other nodes to get aware of which tree to be used, the packets should carry a mark that identifies the correct tree. A node that detects a failure, attempts to use the tree with the ID one higher than the tree ID in the incoming packets. If the maximum ID is reached without finding a viable next hop, packets will be dropped. We count an unsuccessful sample when this occurs.

IPRT builds destination-based trees that implicitly give loop-free sequences of backup next hops. IPRT solves the last hop problem in that the redundant trees will reach the last hop through different links. IPRT prevents loops from unrecoverable failures by dropping packets with maximum tree ID.

Multiple Routing Configurations with Deflection (DMRC) [8, 10]

DMRC is based on building a set of logical backup topologies from the original topology. These topologies are built so that each link and node from the original topology does not carry traffic in one of the logical backup topologies. The backup topologies are then given as input to the shortest path calculation, and a next hop is found for each destination in each topology. Each backup topology isolates more than one link and node, so the number of backup topologies needed is not necessary very high. A node that detects a failure will start routing the packets on a backup topology. By marking the topology ID in the packet header, other nodes will know what topology to use, and hence no loops will be created. The node detecting a failure attempts to use the topology with an ID one higher than the topology ID of the incoming packet. When the topology ID has reached the maximum ID and no valid next hop has been found, packets are dropped. We count an unsuccessful sample when this happens. In the networks used for evaluation in this paper, the number of backup topologies for DMRC has been fixed to 5.

DMRC provides loop-free sequences of backup next hops since these are derived from a common topology view using shortest path routing. The last hop problem is solved by using deflection as described in [8]. Since nodes drop packets when the maximum topology ID is reached, DMRC prevents looping when there are failures that cannot be handled.

4 Evaluation

In section 2, we discussed the functional requirements and challenges to the local fast reroute schemes as presented in section 3. In the following, we will evaluate the different schemes with respect to the performance metrics listed in section 2. First, we present results from simulation studies regarding recovery success and path lengths. Then, we evaluate other performance metrics that do not rely on simulations.

4.1 Simulation studies

4.1.1 Method

The goal of our simulation studies has been to capture some general properties of a large set of fast reroute schemes. We have therefore implemented the schemes in our own java-based simulator that outputs the success of each source-destination communication and the path length of the successful communication. Input to this routing simulator is given by a python-based simulator that generates failure scenarios derived from various mobility models or other fault models like node crash. These failure scenarios are given as input to the routing simulator that evaluate how each of the schemes handles a particular failure scenario. We do not consider particular radio technologies or traffic types. We only evaluate how the different schemes are able to maintain connectivity between node pairs and the path lengths between them.

All fast reroute schemes, except the deflection scheme, calculate the backup next hops based on a common original full topology derived from a link state routing protocol (e.g. OLSR). This topology has been input topology to the python simulator. The simulator then produce a set of 100 samples for each failure scenario. One example scenario can be two overlapping link breaks as a result of nodes moving according to a mobility model. When we present figures with 1 to 10 link breaks on x-axis, there are 100 samples for each number of failures, i.e. 1000 in total. Note that we only produce failure scenarios where the topology is not disconnected.

Fault models In general, we model faults as either node failures or link failures. When we evaluate the schemes with regard to *node failures*, we let nodes fail randomly. This also means that all paths transiting the nodes will be affected.

We model *link failures* as disappeared radio connection between neighboring nodes caused by mobility. We have used the Random Waypoint (RW) and

Table 1. Scenarios - parameter settings

	Scenario A	Scenario B
Number of Nodes	16	32
Radio range	218m	218m
Simulated area	612m x 612m	894m x 894m
Speed - min	2m/s	2m/s
Speed - max	60m/s	60m/s
Pause - min	0s	0s
Pause - max	10s	10s
Boundary policy (RD)	Reflection	Reflection
Distance - min (RD)	200m	200m
Distance - max (RD)	600m	600m

Random Direction (RD) mobility models. They complement each other in that they give different position probability in the simulated area. RW shows a tendency to arrange nodes closer to the center than RD [2]. RW has the advantage of being a simpler model with fewer input parameters, but it has also known problems [24], and one might consider our results derived with the RD model as more reliable than those derived with the RW model.

Table 1 gives a summary of the most important parameter settings that have been used for scenario A and B in the mobility trace generator. The settings have been used for both the Random Waypoint model and the Random Direction model, except “Distance” and “Boundary policy” that are only input parameters to the Random Direction model. We have also tested other scenarios, but due to space limitation we only present the ones in table 1.

These failure scenarios are then given as input to the routing simulator, which outputs recovery success and path lengths for all the fast reroute schemes. The figures presented for recovery success and path lengths show the average with 95th percentile.

4.1.2 Recovery success

Figures 1 to 5 present some of the results regarding recovery success. With the red frame we will draw the most attention to the scenarios with few link breaks. This is because a proactive link-state based routing scheme would most likely be used in scenarios with less probability for failures. However, we show up to ten link breaks to illustrate the tendency in extreme cases.

It is clear that all recovery schemes improve the success rate compared to the normal case without any rerouting. We have evaluated the schemes in two modes; configured for link failures only and configured for combined node and link failure recovery. Figures 1 and 2 show these configuration strategies for the same link failure scenarios. All schemes except LFA gives as good as the same recovery success when configured for both node and link failures as for link failures only.

LFAs gives lower recovery success due to stricter condition for loop-freeness. In figure 5, we see that the schemes also give good recovery success when the failure scenario is node breaks.

Figures 2 and 3 illustrate the difference between the RD and RW mobility models. The fact that RW gives very centered nodes results in some better recovery success. This is particularly true for deflection. Another observation is that deflection gives a stable high success rate in most scenarios and even with many failures. There are two main reasons for the high success rate of deflection. First, deflection is the scheme that can take the biggest advantage of new links that emerge due to mobility. A node that detects a failure can deflect to all existing links. Second, the deflection scheme does not drop packets until for instance a time to live (TTL) expires. FIFR shares this second property with deflection, meaning that it will try to recover until a TTL expires. FIFR on the other side cannot send packets to any available link since the backup next hops are pre-planned. The delayed packet dropping for both deflection and FIFR has effect on the recovery success. Figure 4 shows the most evident consequence of the delayed dropping. In this scenario, all schemes show lower success rate than in scenario B (figure 3) due to lower average node degree (a factor of about 1.4). The almost unlimited amount of recovery attempts for FIFR and deflection will show higher improvement over other schemes when the number of backup alternatives is lower.

Although, the recovery rate increases with the high amount of recovery attempts, the negative effect with looping may rule out FIFR and deflection as candidates. With 4 link breaks in figure 4, about 12 % of the traffic will be looping around in the network consuming resources.

DMRC and IPRT share some similarities with deflection and FIFR as they also can attempt to recover more than once; DMRC can try all topologies before giving up and IPRT can try both trees. However, they prevent lasting looping since they drop packets when maximum topology or tree ID has been reached.

Not-via has the disadvantage that it will never try to recover a second failure when it is already in a Not-via tunnel.

For LFAs, there may be cases where loop-free alternates did not exist in the original topology. If a node detects a failure and there is no LFA, the packets are dropped. This is also the case if the LFA has disappeared due to node mobility. If there are more than one failure in the network and there exist LFAs for these failures, the nodes will attempt to reroute and packets may loop between the failures.

Since proactive link-state based IP routing would be used in fairly stable networks, the recovery success for few failures should get the most attention. The schemes that guarantee single failure recovery give good recovery with few failures in most scenarios. In the scenario in figure 4, the recovery success is lower; however, the improvement compared to the normal case without recovery is still good. Deflection and LFAs do not give any guarantees even for one failure. We have also discussed how particularly FIFR and deflection may cause unlimited looping for packets that cannot be recovered.

4.1.3 Path lengths

Path lengths or hop count is a very important metric for most ad hoc network technologies that exist today [7]. This is due to the broadcast nature of wireless communication. An extra hop does not only affect the two neighbors communicating, but all nodes that can hear the broadcasting. [4] shows that the throughput in the network decreases close to exponentially with the number of hops in an 802.11 network. This also demonstrate the severe effect of the looping discussed with FIFR and deflection in section 3. We must add that the penalty of extra hops may be less severe in other and future technologies; however higher penalty than in wired switched networks will always be the case.

Figures 6 and 7 present some of the results from the path length evaluations. The main conclusion from these results is that there is a strong correlation between success rate and path lengths in that higher success rates give longer paths. Deflection gives considerably longer paths and serves as an extreme example of this correlation. Another observation is that IPRT seems to provide some longer paths than what could be expected from the recovery success. FIFR on the other hand, seems to provide some shorter paths than what could be expected. The results show path lengths for the communication samples that are successful, and hence they do not reflect the loops associated with deflection and FIFR and to some extent LFAs.

Figures 6 and 7 show the same failure scenario with the schemes configured for link failures and both node and link failures, respectively. We see that planning for both node and link failures does not give longer recovery paths than planning for link failures only.

If we give the most attention to few failures, we see that the difference between the schemes is not very prominent. Therefore we could say that path length is not the most critical factor when selecting the most appropriate scheme. It is much more important to consider the penalty from packet looping with deflection, FIFR and LFAs.

4.2 Other performance factors

In section 2, we listed other performance factors than recovery success and path lengths that may effect the selection of the most appropriate scheme. In the following, we discuss these.

4.2.1 State requirements

To take advantage of IP fast reroute, the nodes in the network must store extra information about backup next hops. With deflection, no extra state must be stored as the backup next hop is selected randomly. LFA requires that the nodes store a backup next hop for the destinations that have an LFA, i.e. at most 1 backup next hop for each primary next hop ($1 \cdot normal$). The Not-via approach requires the nodes to store one extra next hop per Not-via address. Since there will be one address for each end point of a link, the number of extra state will be $2 \cdot e$, where e is the number of links in the network. Since FIFR decides the next hop based on what neighbor the packet come from, it needs to store information about the neighbors. It does not require more next hops because it does not store a next hop for each destination for each neighbor. IPRT nodes need to store next hops for each tree, i.e. the backup next hops require $2 \cdot normal$ extra next hop state. DMRC requires the nodes to store next hops for each backup topology, i.e. the extra next hop state will be $n \cdot normal$ where n is the number of backup topologies.

4.2.2 Computation complexity

All the schemes except deflection need to calculate the backup next hop based on a common topology view. The complexity of this calculation may be an important issue with computation weak nodes. The difference between the different schemes are not very prominent. If v is the number of nodes, e the number of links, n the number of backup topologies and Δ the avarege node degree, the complexity will be as follows. LFA can be calculated in $\mathcal{O}(v \cdot logv)$. Not-via would need $\mathcal{O}(e^2 \cdot \log v)$. FIFR would need $\mathcal{O}((\log v)^2 \cdot e)$. IPRT would need $\mathcal{O}(v \cdot e)$. DMRC would need $\mathcal{O}(v \cdot e \cdot n \cdot \Delta)$.

4.2.3 Forwarding complexity

The forwarding complexity of the schemes can influence on the forwarding delay in a node. This is particularly true if the nodes have limited processing power. Deflection adds little complexity to the forwarding since a backup next hop is just selected randomly on the fly.

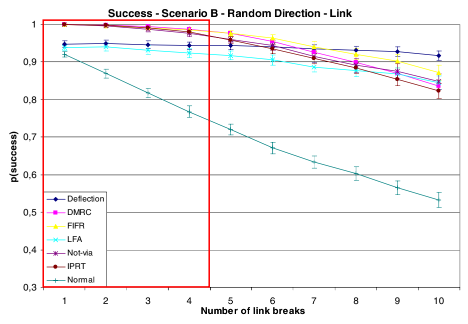

Figure 1. Recovery success for link failures and Random Direction with scenario B. Schemes are optimized for link failures.

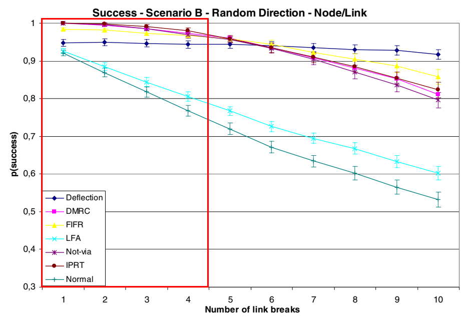

Figure 2. Recovery success for link failures and Random Direction with scenario B. Schemes are optimized for both node and link failures.

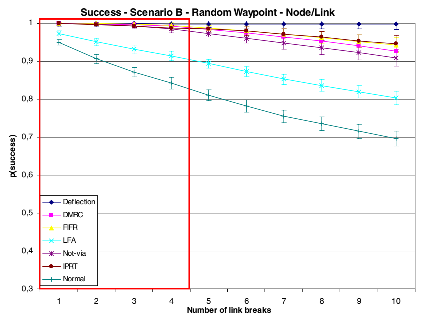

Figure 3. Recovery success for link failures and Random Waypoint with scenario B. Optimized for both node and link failures.

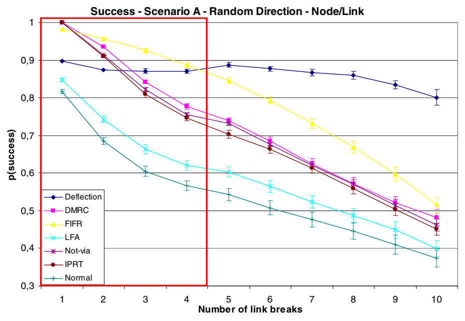

Figure 4. Recovery success for link failures and Random Direction with scenario A. Optimized for both node and link failures.

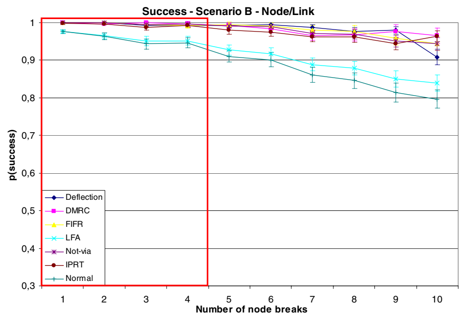

Figure 5. Recovery success for node failures with scenario B. Optimized for both node and link failures.

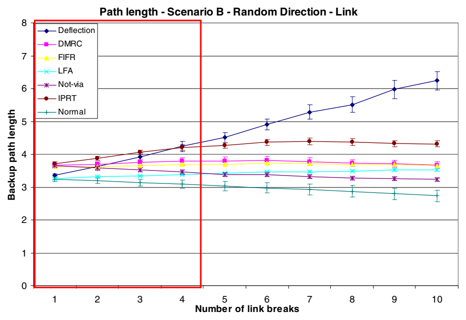

Figure 6. Path lengths for link failures and Random Direction with scenario B. Optimized for link failures.

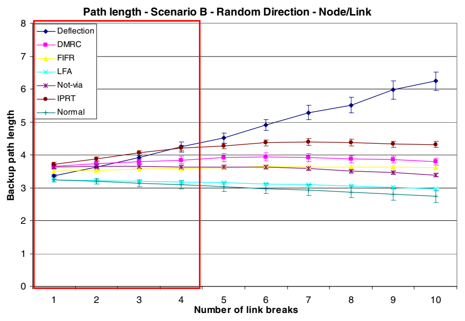

Figure 7. Path lengths for link failures and Random Direction with scenario B. Optimized for both node and link failures.

LFA requires that a node looks up the LFA if there exists one. Not-via requires a node to look up the Not-via address for the failure and to encapsulate the packet into a new header. The tunnel end point must decapsulate the tunnel header. This tunneling may also impose other negative effects if the extra header make the packet exceed the maximum transfer unit (MTU). Then packet fragmentation can add even more complexity to the forwarding process. FIFR adds complexity in that all nodes (not only the detecting node) need to know what neighbor node it received a packet from. Currently, there is no support for this on the IP layer. Potential solutions could be notification from lower layers or previous hop address in the packet header. IPRT adds complexity in that a node detecting a failure must mark packets with the treeID. The other nodes must then look up next hop based on both destination address and the treeID. DMRC adds the same complexity as IPRT, using backup topologies instead of trees.

It is clear that Not-via and FIFR add far more forwarding complexity than the other schemes. This may influence the forwarding delay and throughput in the network.

4.2.4 Flexibility for optimizations

IPRT and DMRC distinguish themselves from the other schemes in their flexibility for optimizations. Particularly, this is true for multi-failure extensions and load balancing or QoS routing for the recovery traffic. The reason for this that the backup routing is based on using logical sub-topologies (trees and backup topologies) of the full topology. These sub-topologies can be built to optimize selected performance aspects [8, 11, 20, 23].

4.3 Evaluation summary

Table 2 summarizes how the different schemes fulfill the functional requirements, whether they solve the challenges and how they perform with respect to important performance metrics as presented in section 2. In the following we point out the most critical metrics for selecting the most appropriate scheme.

All schemes offer local recovery from the node that detects the failure in a proactive manner without diagnosing the root cause of failure. We also regard single failure guarantees as an important feature in most scenarios where we would use proactive routing. Deflection and LFA cannot provide this guarantee.

If there are more failures than the scheme can handle, we regard it as very important to prevent looping. Else, traffic will loop and consume valuable radio resources. Deflection, FIFR and to some extent LFA are not able to prevent this looping. In wireless networks, we find this looping so severe that these schemes can not be considered candidates the way they are designed today.

Good recovery success rate during multiple failures and moderate path lengths are also important features in scenarios demanding high dependability. Most of the schemes give almost the same recovery success and the path lengths are mostly correlated with the success rate. Therefore, these metrics do not influence much on the selection of the most appropriate scheme.

State requirements are often regarded important in Internet routers that have to store next hop information for millions of IP addresses. We assume that our scenarios will be based on local routing, and since the state requirements for the different schemes do not differ considerably, we do not use this metric to select schemes.

Computation complexity could be a critical metric if the nodes are very computation weak. In our scenarios we do not consider such nodes, also due to the fact that we are using proactive routing, which in itself gives frequent calculations. Nevertheless, the schemes do not differ much with regard to this metric.

Forwarding complexity will most likely be an important metric in most scenarios. Complex decision process, packet inspection and packet modification will influence on the forwarding speed, and it may impose some delay. We regard Not-via as to complex due to tunneling and potential fragmentation of packets. Also FIFR may be to complex since there are currently no support for forwarding based on previous neighbor knowledge. Both DMRC and IPRT add packet re-marking as a task to the node that detects the failure. This may also influence on the forwarding speed. However, it is a less complex process than for Not-via and FIFR.

Flexibility for optimization is regarded as a nice feature, but not critical for the selection of the most appropriate scheme for our scenarios.

According to the selection of most critical metrics we find that Deflection and LFA are removed from the list of candidates due to lack of guarantees for one failure tolerance and looping problems during multiple failures. Also FIFR can be removed as candidate due to the same looping issues during multiple failures. We also find Not-via very complex when it comes to forwarding. Although not perfect, it seems like our evaluations point out IPRT and DMRC as the most viable schemes to solve fast reroute in our scenarios with proactive routing.

5 Conclusion and future work

In this paper, we have evaluated a set of IP fast reroute schemes for scenarios with proactive link-state routing. According to our best knowledge, no similar schemes have currently been developed for wireless networks. In addition, this is the first paper that gives such a thorough evaluation of all these candidate schemes. The results show that the source-destination connectivity improves considerably compared to having no fast reroute. We have also seen that planning for both node and link failures can be beneficial. Taking all the functional requirements and important performance metrics into consideration, we have found that IPRT and DMRC is the most viable candidates for IP fast reroute in the scenarios addressed in this paper.

Although IPRT and DMRC seems to be the best candidates, they are not perfectly designed to fit any wireless network that uses proactive routing. Currently, they both rely on full topology knowledge to calculate the alternative next hops. As a future task it is required to optimize their performance in cases where only limited neighbor information is provided. In addition, one should also address the case where the different nodes have inconsistency in their topology view. There is also a great potential for improving the state requirements and forwarding complexity of both schemes.

References

[1] A. Atlas and A. Zinin. Basic specification for IP fast reroute: Loop-free alternates. Internet Draft, 2005. draft-ietf-rtgwg-ipfrr-spec-base-04.txt.

[2] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless*

Table 2. Evaluation summary

	Deflection	**LFA**	**Not-via**	**FIFR**	**IPRT**	**DMRC**
Local	Yes	Yes	Yes	Yes	Yes	Yes
Proactive	Yes	Yes	Yes	Yes	Yes	Yes
Root cause of failure	Yes	Yes	Yes	Yes	Yes	Yes
Single link and node protection	No	No	Yes	Yes	Yes	Yes
Last hop problem	Yes	Yes	Yes	No	Yes	Yes
Loopfree sequences of next hops	No	Yes	Yes	Yes	Yes	Yes
Loop avoidance during not recoverable failures	No	No	Yes	No	Yes	Yes
Recovery success rate -rank	1	6	5	2	4	3
Path lengths - rank	6	1	2	3	5	4
State requirements	0	$1 \cdot normal$	$2 \cdot e$	0	$2 \cdot normal$	$n \cdot normal$
Computation complexity	0	$\mathcal{O}(v \cdot logv)$	$\mathcal{O}(e^2 \cdot \log v)$	$\mathcal{O}((\log v)^2 \cdot e)$	$\mathcal{O}(v \cdot e)$	$\mathcal{O}(v \cdot e \cdot n \cdot \Delta)$
Forwarding complexity - rank	1	2	6	5	3	4
Flexibility for optimizations	No	No	Poor	No	Good	Good

Communications & Mobile Computing (WCMC): Special Issue on Mobile Ad Hoc Networking: Research, Trends, and Applications, 2(5):483–502, Aug. 2002.

[3] M. Chandra and A. Roy. Extensions to OSPF to Support Mobile Ad Hoc Networking. IETF Internet Draft, 2007. draft-chandra-ospf-manet-ext-04.

[4] J. L. Charles, B. Douglas, S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *ACM International Conference on Mobile Computing and Networking*, Rome, Italy, July 2001.

[5] T. Cicic, A. F. Hansen, and O. K. Apeland. Redundant trees for fast IP recovery. In *To appear in IEEE Broadnets 2007*, North Carolina, US, 2007.

[6] T. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR). IETF RFC 3626, 2003.

[7] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *International Conference on Mobile Computing and Networking*, Philadelphia, PA, USA, 2004.

[8] A. F. Hansen, O. Lysne, T. Čičić, and S. Gjessing. Fast Proactive Recovery from Concurrent Failures. In *ICC 2007*, June 2007.

[9] K. Holter, A. Hafslund, F. Li, and K. Ovsthus. Design and implementation of wireless OSPF for mobile ad hoc networks. In *Scandinavian Workshop on Wireless Ad-hoc Networks (ADHOC 06)*, Stockholm, Sweeden, May 2005.

[10] A. Kvalbein, A. F. Hansen, T. Čičić, S. Gjessing, and O. Lysne. Fast IP network recovery using multiple routing configurations. In *IEEE INFOCOM*, Apr. 2006.

[11] A. Kvalbein, T. Čičić, and S. Gjessing. Post-failure routing performance with multiple routing configurations. In *Accepted for IEEE INFOCOM*, 2007.

[12] S. J. Lee and M. Gerla. AODV-BR: backup routing in ad hoc networks. In *Wireless Communications and Networking Conference*, Chicago, IL, USA, 2000.

[13] S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *IEEE ICC*, Helsinki, 2001.

[14] R. Leung, J. Liu, E. Poon, A.-L. C. Chan, and B. Li. MP-DSR: A qos-aware multi-path dynamic source routing protocol for wireless ad-hoc networks. In *IEEE International Conference on Local Computer Networks (LCN'01)*, 2001.

[15] M. K. Marina and S. R. Das. Ad hoc on-demand multipath distance vector routing. *Wiley Wireless Communications and Mobile Computing*, 6(7):969–988, 2006.

[16] R. Ogier, F. Templing, and M. Lewis. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). IETF RFC 3684, Feb. 2004.

[17] G. Parissidis, V. Lenders, M. May, and B. Plattner. Multi-path routing protocols in wireless mobile ad hoc networks: A quantitative comparison. In *NEW2AN 2006*, 2006.

[18] M. Pearlman, Z. Haas, P. Scholander, and S. Tabrizi. Alternate path routing in mobile ad hoc networks, 2000.

[19] M. Radimirsch. HIDENETS use-case scenarios and preliminary reference model. HIDENETS consortium, Deliverable D1.1, 2006. www.hidenets.aau.dk/Public+Deliverables.

[20] S. Ramasubramanian, H. Krishnamoorthy, and M. Krunz. Disjoint multipath routing using colored trees. *Elsevier Computer Networks Journal*, 2006.

[21] M. Shand and S. Bryant. IP Fast Reroute Framework. IETF Internet Draft, Oct. 2006. draft-ietf-rtgwg-ipfrr-framework-06.txt.

[22] J. Tsai and T. Moors. A review of multipath routing protocols: From wireless ad hoc to mesh networks. In *ACoRN Early Career Researcher Workshop on Wireless Multihop Networking*, July 2006.

[23] G. Xue, L. Chen, and K. Thulasiraman. QoS issues in redundant trees for protection in vertex-redundant or edge-redundant graphs. In *Proc. ICC*, volume 5, pages 2766–2770, Apr. 2002.

[24] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. *INFOCOM 2003*, 2:1312–1321 vol.2, 30 March-3 April 2003.

[25] 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 *Proceedings of IEEE Global Internet*, Mar. 2005.