Fast Recovery from Link Failures in Ethernet Networks

A. PRATAP NAIDU
M. Tech Department of Computer Science Engineering (CN & IS)
Tirupati (India)

M.V.R MANEESHA
M. Tech Department of Computer Science Engineering
Tirupati (India)

Abstract:
Fast-recovery from link failures is a well-studied topic in IP networks. Employing fast-recovery in Ethernet net- works is complicated as the forwarding is based on destination MAC addresses, which do not have the hierarchical nature similar to those exhibited in Layer 3 in the form of IP-prefixes. Moreover, switches employ backward learning to populate the forwarding table entries. Thus, any fast recovery mechanism in Ethernet networks must be based on undirected spanning trees if back-ward learning is to be retained. In this paper, we develop three alternatives for achieving fast recovery from single link failures in Ethernet networks. All three approaches provide guaranteed recovery from single link failures. The approaches differ in the technologies required for achieving fast recovery, namely VLAN rewrite or mac-in-mac encapsulation or both. We study the performance of the approaches developed on five different networks.

Keywords: *Backward learning, Ethernet, Fast recovery, Independent spanning trees, Link failure, Network availability*

---

1. Introduction

ETHERNET is becoming an attractive solution in metropolitan and wide area networks as it offers a cost-effective way to provision high data rate services. However, the simplicity and cost - effectiveness come with two major shortcomings: poor support for traffic engineering, and slow failure recovery times. These two shortcomings are a direct consequence of employing an undirected spanning tree as the basis for forwarding. The spanning tree plays a key role in:

1. Reducing the unnecessary overhead created by broadcasting when a destination address is not available,
2. Retaining the backward learning mechanism, which is crucial in supporting the scalability and mobility of the end-hosts? The spanning tree, however, provides only one path between any node pair, and hence the failure of any link or node would disconnect the spanning tree.

To overcome the deficiencies of the spanning tree approach, there have been several revisions to the original spanning tree protocol, such as support for faster re-convergence (RSTP) and support for multiple spanning trees (MSTP) that can help create smaller regions for recovery. Protocols to reduce fault detection times, such as bi-directional forwarding detection (BFD) were developed. Despite all these efforts, we still lack a fundamental understanding of the application of undirected spanning trees in achieving good resiliency in network design.

In this paper, our goal is to study the use of multiple spanning trees with interesting properties for achieving fast recovery in Ethernet networks. We develop methods to achieve fast recovery from link failures in VLANs using proactive approaches that rely only on local information with a constant overhead. Every spanning tree may be configured with a unique VLAN identifier. The VLANs are precompiled and preconfigured, thus enabling fast recovery from link failures. In addition, traffic may be split over multiple VLANs to provide increased cross-sec tonal bandwidth. The algorithms and protocols have provable performance guarantees.

A. Fast Recovery in IP vs. Ethernet

Fast recovery from link and node failures has been studied extensively in the context of IP networks. Tradition- ally, IP routing table entries are computed by constructing destination rooted trees for IP prefixes. The trees for different IP prefixes are computed separately from each other. The destination-rooted trees are directed in nature, directed towards the destination. Thus, the routes between two IP endpoints may not necessarily be symmetrical. Fast recovery in IP networks is achieved by providing one or more backup ports, in addition to the primary forwarding port used under no failures. The for- warding of IP packets is then based on the destination IP ad- dress, and some additional information, which is either carried in the packet or derived from the incoming port on the router.

B. Related Work

An undirected spanning tree gets disconnected by a link or node failure. The spanning tree protocol and its variants re- cover from failures by computing another tree after the link failure. The convergence time of the original spanning tree algorithm was on the order of 30 to 50 seconds. RSTP, which was developed later in 802.1d, can reduce the convergence time anywhere from tens or hundreds of milliseconds to a few seconds depending on considerations such as net- work topology, port manipulation times, time for failure detection,

Our goal in this paper is to employ multiple spanning trees, each identified with a unique VLAN, to achieve fast recovery from link failures. By fast recovery, we refer to the forwarding of packets along an alternate port (path) from the switch connected to the failed link. The switching of traffic to an alternate path implies changing the VLAN tag in the packet, thereby employing a different spanning tree for routing. In the authors employ multiple spanning trees over which traffic is distributed. Upon a failure, an intermediate switch would choose a spanning tree whose VLAN identifier is higher than the one that failed. The papers, however, do not quantify the number of spanning trees needed to guarantee a single link failure recovery, or even mention how the trees are computed. Link- disjoint spanning trees have been proposed as a mechanism for achieving load balancing and failure recovery. In these approaches, two spanning trees are constructed such that no undirected link is common to both trees. The use of link-disjoint spanning trees in conjunction with switching traffic from one spanning tree to another at the point of failure guarantees recovery from single link failures. However, a network has to be four edge connected to obtain two link-disjoint spanning trees which is a stringent requirement. Note that, in contrast, we can compute two destination-rooted link-independent spanning trees1 in two edge connected networks, and achieve fast rerouting in IP networks.

The use of multiple spanning trees, each identified using a unique VLAN tag, is a popular approach for distributing traffic across different paths. There have been prior works that have looked at the issues of improving fault tolerance and bandwidth guarantees. Works have studied how one could re-structure the original spanning tree upon link failures so that the changes during re-convergence are mitigated to some extent. Such schemes often rely on messaging or might require the use of several backup trees which cannot scale well. Although these schemes can re-connect the broken tree quickly (termed fast recovery or fast re-connection), it is not obvious how the packets at the switch can be re-forwarded until such a new tree is set up. Thus, there will be packet losses, and control messages being exchanged with other switches until a new tree is setup.

Table 1

Approach	Connectivity	# VLANs	Encap (#)
3Trees	3E	3	No
2Trees	2E	2	Yes (1)
1Tree	2E	2	Yes (2)

A Comparison of Network Connectivity, Number of VLANS Required, And 802.1ah MAC-IN-MAC Encapsulation Requirements for the Different Approaches Developed in This Paper topology, the techniques to be employed for forwarding upon encountering a failure, and the number of VLAN tags required.

C. Contributions

Our contributions in this paper are as follows. We develop three approaches, namely the 3Trees, 2Trees, and 1Tree that each provides different tradeoffs. All approaches are based on multiple spanning trees and shifting the packet from one spanning tree to another upon failure, thus requiring VLAN re-write capability at the switches. In addition, some of the approaches may require MAC-in-MAC encapsulation 802.1ah with either 1 or 2 levels of encapsulation. Table I shows the summary of the requirements for the different approaches developed in this paper.

All three solutions guarantee recovery from single link failures, and the forwarding mechanism upon a failure is based purely on local information. Moreover, all of the solutions are proactive approaches. Thus, the switches may be programmed with the actions to be taken for forwarding a packet under normal circumstances and upon failures.

We extend the contributions made in the conference version of this paper by providing proofs for theorems, additional performance evaluation, discussion of new results, and detailed illustrations of the proposed schemes.

D. Organization

In Section II, we describe issues that are common to any fast recovery approach that employs multiple spanning trees. In Sections III, IV, and V we describe the 3Trees, 2Trees, and 1Tree approaches, respectively. We illustrate the working of each approach with examples. In Section VI, we evaluate the performance of these approaches. We conclude the paper in Section VII. The Appendix describes the necessary theorems and proofs required to ensure the correctness of the 1Tree approach.

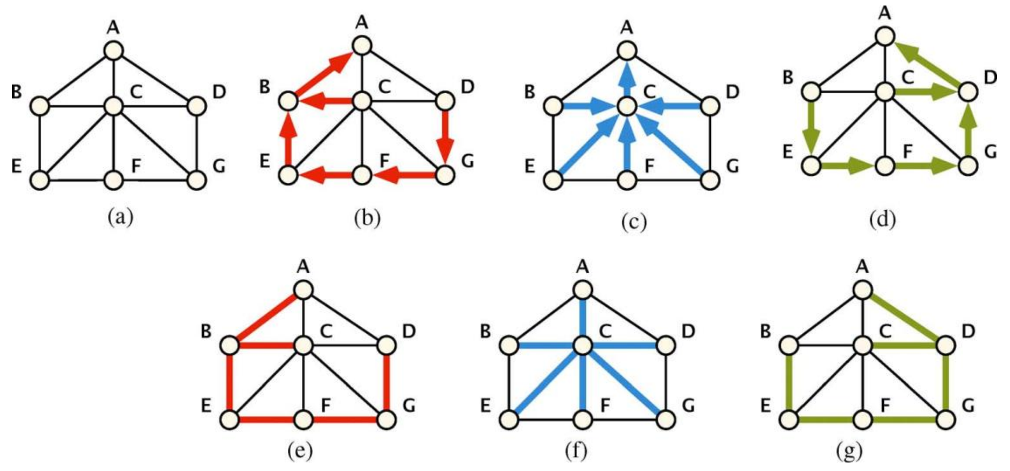
2. Fast recovery with multiple spanning trees-general considerations

Recall that any solution for fast recovery in Ethernet net- works needs to use an undirected spanning tree due to back- ward learning. One approach to guarantee single link failure recovery is to decompose the network into multiple spanning trees such that, for any given link failure, there exists a spanning tree that doesn't contain that link. When a failure occurs, the switch connected to the failure may simply transfer the packet to the spanning tree that does not contain the failed link by rewriting the VLAN tag in the packet. Thus, every switch must have the knowledge of which VLAN to rewrite the packet with, upon a link failure. There exists a spanning tree that doesn't contain that link. When a failure occurs, the switch connected to the failure may simply transfer the packet to the spanning tree that does not contain the failed link by rewriting the VLAN tag in the packet. Thus, every switch must have the knowledge of which VLAN to rewrite the packet with, upon a link failure.

One approach to use multiple spanning trees is to designate one tree as primary and the others as backup. Thus, when a link in the primary tree fails, the switch connected to the failed link would rewrite the VLAN tag in the packet to correspond to the secondary tree and forward it along the secondary tree. However, there is a drawback to this approach. As the switches populate the forwarding table entries using backward learning, the switches would not have learnt about the end hosts on the backup spanning tree. This limitation would force the intermediate switch to broadcast the packet on the backup spanning tree. However, if the traffic is spread over all the spanning trees, then the broadcasting upon a failure may be avoided. If the traffic is sent over all the spanning trees, then the switches would need a way to verify if the incoming packet has already encountered a failure or not. We may encode this information with one bit. We may use one bit from the 3-bit class-of-service (CoS) field in the VLAN header. Note that the above considerations are applicable to any fast recovery mechanism using multiple spanning trees, thus they naturally apply to the approaches developed here.

3. 3trees Approach

Our first approach is for three edge connected networks. We employ three spanning trees, referred to as the red, blue, and green, identified using three VLAN tags. Every link will be present in at most two of the trees by construction. Thus, when a link fails, the switch connected to the failed link can re-write the VLAN corresponding to the tree on which the link is not present.



Procedure for forwarding in 3Trees approach

Given: Destination d, VLAN tag of packet $T \in \{R, B, G\}$.
n_d^T denotes the next-hop on VLAN T to reach destination switch d.

```
1) Forwarding:
    a) If link to $n_d^T$ is available, forward packet. Go to Step 4.
2) Check if packet has already seen a failure:
    a) If $T \neq R$, the packet has seen a failure.
    b) Drop packet, go to Step 4
3) Recovery:
    a) If link is present on blue VLAN,
        i) Re-write VLAN tag in packet to $G$.
        ii) Go to Step 1.
    b) In all other cases,
        i) Re-write VLAN tag in packet to $B$.
        ii) Go to step 1.
4) Stop.
```

Fig. 1 forwarding procedure in 3Trees approach

The 3Trees approach is designed for three edge connected networks. Although the 3Trees approach requires only VLAN re-write capability at the switches, the three edge connectivity requirement may not be met by some real-life networks. In addition, the approach requires the use of three trees, hence three VLAN tags per virtual network. As the number of VLAN tags available is only 4096, allocating three VLAN tags per network may not be preferred. For these two reasons, it is desired to have fast recovery methods that work on two edge connected networks, and preferably with fewer number of VLAN tags per network. In the following sections, we develop two approaches to achieve this goal.

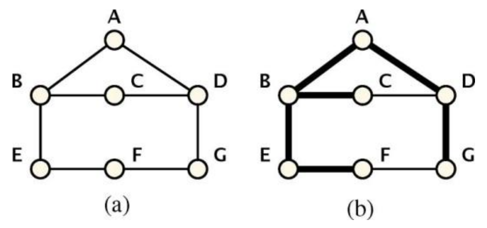
4. 2trees Approach

In an arbitrary two edge connected network, we may require spanning trees such that for every link, there exists at least one spanning tree that does not contain the link. Spanning trees with this property are required only if we are restricted strictly to using VLAN re-write as the only mechanism for fast recovery. However, we may achieve fast recovery with only two spanning trees, thus two VLAN tags, if we can employ mac-in-mac en- capsulation in addition to VLAN re-writing.

```
Procedure for forwarding in 2Trees approach
Given: Destination $d$, VLAN tag of packet $T \in \{R, B\}$, One bit information $f$ in CoS field of packet.
 1) Forwarding:
      a) If link to $n_d^T$ is available, forward packet. Go to Step 4.
 2) Check if packet has already seen a failure:
      a) If $f = 1$, the packet has seen a failure.
      b) Drop packet, Go to Step 4
 3) Recovery:
      a) Set the failure bit $f = 1$
      b) If $n_d^R \neq n_d^B$:
           i) Re-write VLAN tag in packet to $B$.
          ii) Go to Step 1.
      c) If $n_d^R = n_d^B \neq n_r^R$:
           i) Re-write VLAN tag in packet to $B$.
          ii) Encap {Destination MAC, VLAN tag, $f$} as {$r$,$R$,1}.
         iii) Go to Step 1.
      d) If $n_d^R = n_d^B = n_r^R$:
           i) Encap {Destination MAC, VLAN tag,$f$} as {$r$,$B$,1}.
          ii) Go to Step 1.
 4) Stop.
```

Fig. 2 forwarding procedure in 2Trees approach

The procedure shown in Fig. details the forwarding actions taken at a switch. Because the 2Trees approach will employ mac-in-mac encapsulations, the destination and tag in the procedure refers to the MAC address and VLAN tag of the outermost header in the packet. Consider the situation when the red forwarding neighbor to reach the root node of the spanning tree is not the same as the forwarding neighbor for. Note that a spanning tree provides a unique path for every node to reach the root. Thus, if the red forwarding neighbors for a destination and root are different at a switch, then it implies the following. (a) The switch can still reach the root on the red tree. (b) The path to the root from on the red tree traverses the intermediate switch, and therefore, the destination is still connected to the root on the blue tree. Thus, we may forward the packet to the root along the red tree, and forward the packet from the root to the destination along the blue tree. To achieve this solution, we employ mac-in-mac encapsulation. We first re-write the VLAN tag on the original packet to that of the blue spanning tree. We then add an encapsulation (outer) header to forward the packet to the root on the red tree.



Construction of a shortest path spanning tree in an example two edge connected network. (a) Example network. (b) Undirected shortest path spanning tree rooted at node A. (a) Example network, (b) Spanning tree.

The 3Trees and 2Trees approaches developed thus far required that the trees be constructed simultaneously. If one de- sires a specific tree to be employed under no failure scenario, then it may not be possible to construct the other tree(s). Consider the example network shown in Fig. (1), and assume that the spanning tree protocol is run with node A being chosen as the root. The resulting spanning tree is shown in Fig. (b). It can be easily seen that a second link-independent spanning tree rooted at node A cannot be constructed as both links of node A are used in the spanning tree.

In the 3Trees and 2Trees approaches, one may use a separate tree for forwarding under the no failure scenario by dedicating another VLAN tag, referred to as the normal VLAN tag. Thus, under this approach, the traffic under the no failure scenario would be forwarded based on the normal VLAN tag. When the packet encounters a failure, the switch would treat the packet as one that was forwarded based on the VLAN tag of the failed outgoing edge, and select the corresponding backup forwarding action. If used with a normal VLAN tag, the 3Trees, and 2Trees approaches three VLAN tags, respectively.

5. 1 tree Approach

To overcome the limitation of the 3Trees and 2Trees approaches, we develop another approach that allows any default tree to be used for forwarding traffic under no failure scenario. We refer to this approach as the 1Tree approach.

The 1Tree approach is based on the technique for backup port assignment for IP fast re-route developed by Xi *et al.*. Using their approach as a starting point, we show that it is possible to construct a collection of trees to recover upon a link failure with mac-in-mac encapsulations. The collection of trees can all use the same VLAN tag. Thus, the 1Tree approach requires two VLAN tags: one VLAN tag to identify the spanning tree used for forwarding when there are no failures, and a second VLAN tag that is shared by a collection of trees. We first briefly describe the ESCAP algorithm introduced in [16]. We then describe how we employ it in our context. Given a two edge connected network, and a primary tree

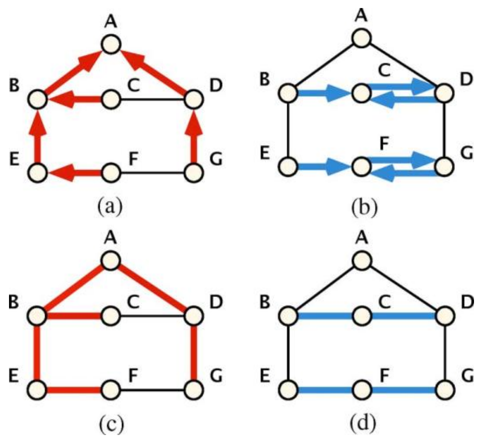
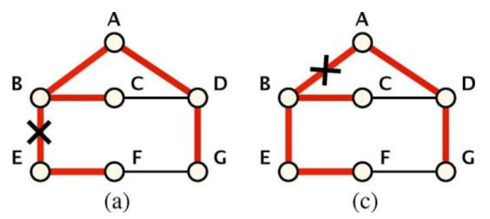


Fig. Construction of backup port assignments according to an ex- ample two edge connected network. (a) Tree rooted at node A. (b) Backup for- warding node assignments. (c) The red VLAN. (d) The blue VLAN. (a) Destination rooted tree, (b) Backup forwarding nodes, (c) Red undirected spanning tree, (d) Blue undirected forest.

Rooted at a destination node, the 1Tree-IP approach computes backup ports for every node. Every node has a primary for- warding neighbor, and a backup forwarding neighbor. In the context of IP networks, the packet forwarding is as follows. 1) If a packet is received from any node other than the primary forwarding neighbor, then the packet is forwarded to the primary forwarding neighbor. 2) If the primary forwarding link is unavailable due to a failure, or if the packet is received from the primary forwarding neighbor, then the packet is forwarded to the backup forwarding neighbor. For a given primary tree rooted at a destination node, the computation of backup other nodes is as follows.



Failure scenarios when the first vlan can be arbitrarily chosen: two failure scenarios depicting the different backup forwarding mechanisms required on the example network. (a) Failure scenario 1, (c) Failure scenario 2.

Procedure for forwarding in 1Tree approach

Given: Destination d, VLAN tag of packet $T \in \{R, B\}$, one bit information f in CoS field of packet.

```
1) Forwarding:
    a) If link to $n_d^T$ is available, forward packet. Go to Step 4.
2) Check if packet has already seen a failure:
    a) If $f = 1$, the packet has seen a failure.
    b) Drop packet. Go to Step 4.
3) Recovery:
    a) Set the failure bit $f = 1$.
    b) If $n_d^R = n_r^R$:
        i) Encap {Destination, VLAN tag, $f$} as $\{z(x), B, 1\}$.
        ii) Go to Step 1.
    c) If $n_d^R \neq n_r^R$:
        [illegible]
        ii) Encap {Destination, VLAN tag, $f$} as $\{z(n_d^R), R, 1\}$.
        iii) Go to Step 1.
4) Stop.
```

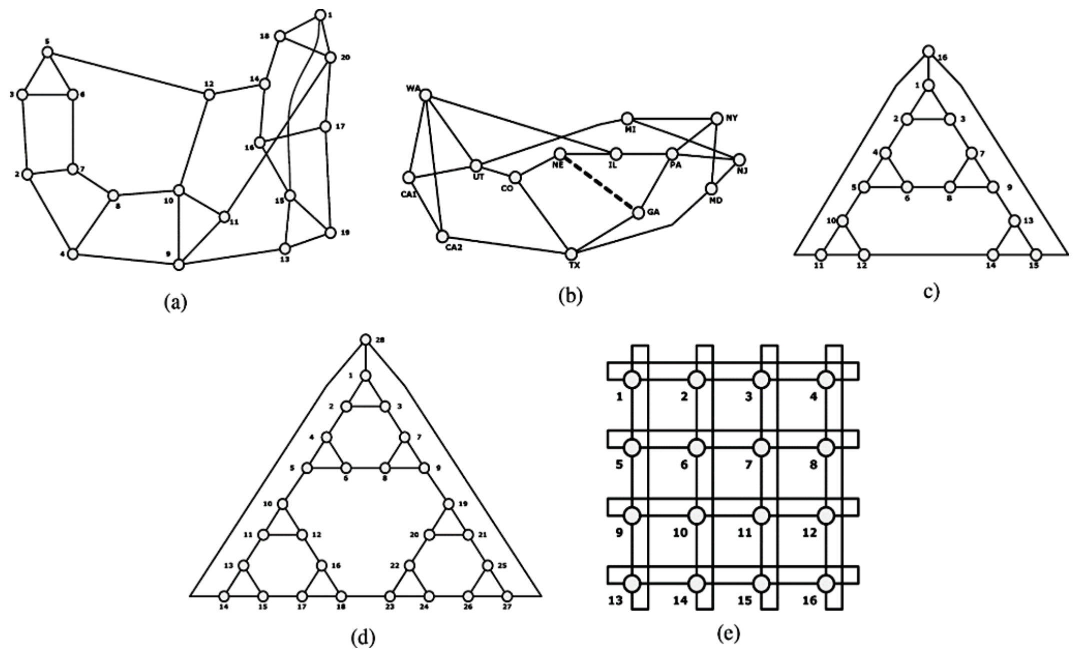


Fig. 3 Forwarding procedure in 1tree approach

Networks considered for performance evaluation. (a) ARPANET (20 nodes, 32 links), (b) NSFNET (14 nodes, 23 links), (c) Node-16 (16 nodes, 24 links), (d) Node-28 (28 nodes, 42 links), (e) Mesh-4 4 (16 nodes, 32 links).

is encapsulated so that it first reaches the exit node of E, which is node G. This encapsulation can be done with an outer blue VLAN tag destined to G. Upon reaching G, the packet is forwarded on the red VLAN towards D, according to the unmodified original header. For example, consider the failure of link A-B shown in Fig. (b). Consider a packet arriving at switch A that is destined to end host attached to switch F. The destination would have been learned over the link A-B on the red tree. As the forwarding neighbor to reach the destination on the red tree, B, is different from that to reach the root A (itself in this case), the packet is encapsulated such that it first reaches the exit node of B, which is switch D, and then reaches switch B. This progress is achieved with an outermost red VLAN tag with destination D, an inner header with destination B on the blue VLAN tag, and the innermost header being the original one to reach node F on the red VLAN left untouched.

6. Performance Evaluation

We evaluate the three approaches developed in this paper: 3Trees, 2Trees, and 1Tree (3T, 2T, and 1T for short) a approaches. We consider five topologies: NSFNET, ARPANET, and three hypothetical topologies Node16, Node28, and Mesh4 4. The first two of these topologies are wide area networks. All the networks considered are three edge connected, with the exception of NSFNET to which we added an additional link to make the network three edges connected. Thus, all three schemes developed in this paper can be employed on each of these topologies for failure recovery. We are interested in the average path length of the approaches under single link failures. This information will help quantify the deviation from the path lengths during normal forwarding in the absence of failures. Studying this deviation is useful because it is representative of the end-to-end delay observed by the destination due to the failure, and thus helps in provisioning buffer capacity for real time applications.

In the 3T and 2T approaches, we simply pick the red tree as the default tree. In the 3T approach, upon link failure, we check to see if the failed link is part of the blue VLAN. If so, we use the green VLAN, and otherwise we simply use the blue VLAN. In case of a link failure when using the 2T approach, the re-forwarding behavior is exactly as is defined in Section IV. In the 1T approach, we use a shortest path tree to mimic the behavior of the spanning tree protocol as the default tree, all these schemes, we first compare the path lengths under no failures to that using optimal shortest paths. We then evaluate the average path lengths under single link failures. Formally, we define the following path metrics that are useful in bench- marking and comparing the various schemes.

A. No Failures

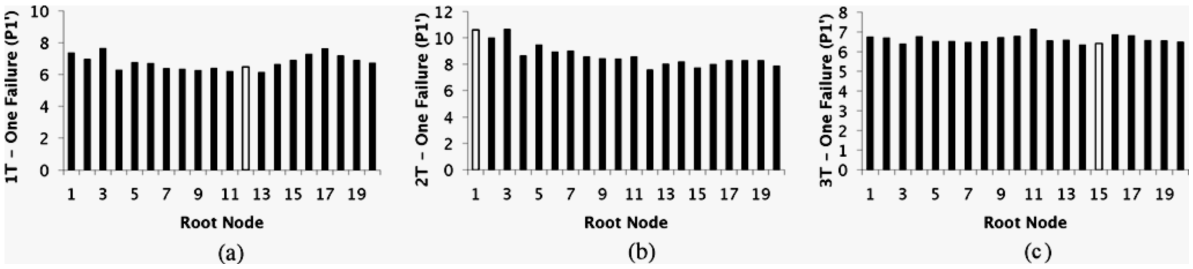
We compute the ideal average path lengths in the network when there are no failures by computing the shortest path between the two nodes in the given network topology. Note that it may not be possible to obtain these average path lengths in reality in Layer-2 networks, as spanning trees are typically constructed with one node as root. Thus, while every node may have a shortest path to the root,

B. One Link Failures

We are interested in computing the average path length when a link failure affects the default path from a source to a destination, which in turn implies only failures of links present on the default spanning tree. All other single link failure scenarios do not affect the re-forwarding schemes, and hence are not considered as they could falsely indicate better performance.

C. Root Node Selection

We highlight that there is a choice to be made for the root, the node at which the tree(s) are constructed in each of the three different approaches, and that it can play a role in the performance of the various approaches. This result is because we do not have the flexibility of constructing the trees per destination in any of the approaches. This result was also alluded to earlier in Section I.A as a key difference between the approaches for Ethernet networks that retain backward learning as opposed to approaches for forwarding in IP networks or Ethernet networks that disable learning.



Impact of root node selection on single link failure performance of the three approaches. (a) ARPANET-1T., (b) ARPANET-2T. & ARPANET-3T

Table 2. Average Backup Path Lengths Using the 1t Approach

	ARPANET	NSFNET	Node-16	Node-28	Mesh-4x4
SPF - No Failure (P_0)	2.79	2.08	2.51	3.57	2.13
SPT - No failure (P_0')	3.76	3.01	3.63	4.93	3.27
% Increase $(P_0' - P_0)/P_0$	35%	45%	45%	38/%	54%
SPF - One failure (P_1)	4.93	3.82	4.68	6.35	3.72
1T - One failure (P_1')	6.48	4.99	6.43	9.32	4.95
% Increase $(P_1' - P_1)/P_1$	31%	31%	37%	47%	33%

D. Performance Results

Before we delve into the performance results of the path lengths on the three approaches, we first briefly study the impact of the root selection. Fig. 10 shows the average path length under single link failures in each of the three approaches on one particular network, ARPANET, for different choices of the root node. The white bar denotes the optimal root chosen according to (7). We observe that, although the root selection has not been optimized for the failure performance, the variance in the average single link failure path length is marginal across different root node selections in all the approaches. We observe similar trends in all other networks as well, but omit the figures for brevity connected, and two header encapsulations (required in the 1Tree approach) may need to be avoided for packet re-forwarding.

Table 3. Average Backup Path Lengths Using the 2t Approach

	ARPANET	NSFNET	Node-16	Node-28	Mesh-4x4
SPF - No Failure (P_0)	2.79	2.08	2.51	3.57	2.13
Red Tree - No failure (P_0')	4.11	3.14	4.17	5.99	3.27
% Increase $(P_0' - P_0)/P_0$	47%	51%	51%	68/%	54%
SPF - One failure (P_1)	4.90	3.94	5.00	7.08	3.72
2T - One failure (P_1')	10.59	6.54	7.53	11.95	7.78
% Increase $(P_1' - P_1)/P_1$	116%	66%	51%	69/%	109%

Table 4. Average Backup Path Lengths Using the 3t Approach

	ARPANET	NSFNET	Node-16	Node-28	Mesh-4x4
SPF - No Failure (P_0)	2.79	2.08	2.51	3.57	2.13
Red Tree - No failure (P_0')	4.19	3.08	3.63	5.10	3.33
% Increase $(P_0' - P_0)/P_0$	50%	48%	45%	43/%	56%
SPF - One failure (P_1)	5.13	3.92	4.68	6.38	3.75
3T - One failure (P_1')	6.42	5.14	6.22	8.89	5.12
% Increase $(P_1' - P_1)/P_1$	25%	31%	33%	39/%	37%

Based on the performance results, we observe that the 3Tree approach provides a good compromise in terms of the path lengths achieved and the overhead involved in fast recovery. Between the 2T and 1T approaches, we observe that the 1T approach provides better path lengths even under single failure scenarios compared to the 2Tree approach. The performance difference is approximately 40% between the two approaches. The tradeoff is in terms of employing two levels of encapsulation in the case of the 1T approach versus one level of encapsulation in the 2T approach.

8. Conclusion

In this paper, we develop three different techniques for achieving fast recovery in Ethernet networks. Two of the approaches (2Tree, 1Tree) are applicable to networks that are at least two edges connected, and one of them (3Tree) is applicable to networks that are at least three edge connected. We show that the approaches provide a tradeoff in terms of path length performance, and the techniques required for achieving fast recovery, such as the number of VLAN tags, dependence on VLAN rewrite, and mac-in-mac encapsulations. Based on the performance results, the approach employing three spanning trees offers a better tradeoff in path length across all the net-works considered in this paper.

References

1. “Metro Ethernet Services-A Technical Overview" (Online) Available: http://www.metroethernetforum.org/Assets/White_Papers/Metro-Ethernet-Services.pdf
2. Gopalan A. and S. Ramasubramanian (2013). “Fast recovery from link failures in Ethernet networks,” in Proc. 9th Int. Conf. Design Rel. Common. Netw., Budapest, Hungary, Mar. 2013, pp. 1–10.
3. M. Ali, G. Chiruvolu, and A. Ge, (2005). “Traffic engineering in metro Ethernet,” IEEE Netw., vol. 19, no. 2, pp. 10–17, Mar.–Apr. 2005.
4. Sofia, R. (2009). “A survey of advanced Ethernet forwarding approaches,” IEEE Common. Surveys Tutorials, vol. 11, no. 1, pp. 91–115.
5. Standard for Local and Metropolitan Area Networks—Rapid Reconfiguration of Spanning Tree, IEEE 802.1w, 2001.