An Optimistic approach to routing protection in IP Networks

Devisetti Sri Balaji
M.Tech(CSE)
GVP College of Engineering
Email:Sribalaji.devisetti@gmail.com

Ms. K.Sudha
Assistant Professor(CSE),
GVP College of Engineering
Email:sudhakapuganti@gvpce.ac.in

MS.M.Sandhya
Assistant Professor(CSE),
GVP College of Engineering
Email:mulagalas@gmail.com

Abstract— **now a day, IP-based networks are becoming popularized to carry all types of traffic, from the traditional best-effort Internet access to traffic with much more stringent requirements such as real-time voice or video services and Virtual Private Networks. As well as routing failures are increased in the same manner and it causes to packet delivery failures. To address the problem, there are lot of fast reroute solutions have been proposed to avoid high packet loss after network fails, but they are worked for specific to single type of routing protocol. It is hard to deploy these solutions together to protect Internet routing including both intra- and inter-domain routing protocols because of their individual computational and storage complexity. For some mission critical services like voice or video over IP, achieving a restoration time in the order of a few tens of milliseconds in IPFRR (IP fast reroute solutions) when the failure occurred. But it is important to reduce that restoration time in order to meet 100% failure coverage. So that In this paper, we propose an optimistic fast reroute solution for routing protection under network failures. Our solution leverages ILP (Integer Linear Program) based p-cycle construction and identifier based direct forwarding to guarantee the effectiveness of routing protection and supports more deployment. In particular, enhanced protection cycle (*e-cycle*) is proposed to construct rerouting paths and to provide node and link protection for both intra- and inter-domain routing protocols. We can our solution by simulations, and the results shows that the solution provides 100% failure coverage for all end-to-end routing paths with approximately two extra Forwarding Information Base entries. Our results prove that the proposed solution effective provides failure recovery and does not introduce processing overhead to packet forwarding.**

I.INTRODUCTION

When a link or node failure occurs in a routed network, there is a period of disruption to the delivery of traffic until the network re-converges on the new topology. Packets for destination that were previously reached by reversing the failed component may be dropped or may suffer looping. Traditionally, such disruptions have lasted for periods of at least several seconds, and most applications have been constructed to tolerate such a quality of service.

Recent advances in routers have reduced this interval to under a second for carefully configured networks using link state IGPs. However, new Internet services are becoming emerging that may be sensitive to periods of traffic loss that are orders of magnitude shorter than this. Addressing these issues is difficult because the distributed nature of the network imposes an intrinsic limit on the minimum convergence time that can be achieved. However, there is an alternative approach, which is to compute backup routes that allow the failure to be repaired locally by the router(s) detecting the failure without the immediate need to inform other routers of the failure. In this case, the disruption time can be limited to the small time taken to detect the adjacent failure and invoke `the backup routes.` This is analogous to the technique employed by MPLS fast-reroute [RFC4090], but the mechanisms employed for the backup routes in pure IP networks are necessarily very different.

II. Problem Analysis

The duration time of the packet delivery disruption caused by a conventional routing transition is determined by a number of factors:

1. The time taken to identify the failure. This may be of the order of a few milliseconds when it can be detected at the physical layer, up to some tens of seconds when a routing protocol Hello is employed. During this period, packets will be unavoidably lost.

2. The time taken for the local router to react to the failure. This will typically involve generating and flooding new routing updates, perhaps after some hold-down delay, and re-computing the router's FIB.

3. The time taken to pass the information about the failure to other routers in the network. In the absence of routing protocol packet loss, this is typically between 10 milliseconds and 100 milliseconds per hop.

4. The time taken to re-compute the forwarding tables. This is typically a few milliseconds for a link state protocol using Dijkstra's algorithm.

5. The time taken to load the revised forwarding tables into the forwarding hardware. This time is very implementation dependent and also depends on the number of prefixes affected by the failure, but may be several hundred milliseconds.

The disruption will last until the routers adjacent to the failure have completed steps 1 and 2, and until all the routers in the network whose paths are affected by the failure have completed the remaining steps.

The initial packet loss is caused by the router(s) adjacent to the failure continuing to attempt to transmit packets across the failure until it is detected. This loss is unavoidable, but the detection time can be reduced to a few tens of milliseconds

In some topologies, subsequent packet loss may be caused by the "micro-loops" which may form as a result of temporary inconsistencies between routers' forwarding tables [RFC5715].

These inconsistencies are caused by steps 3, 4, and 5 above, and in many routers it is step5 that is both the largest factor and that has the greatest variance between routers. The large variance arises from implementation differences and from the differing impact that a failure has on each individual router. For example, the number of prefixes affected by the failure may vary dramatically from one router to another.

In order to reduce packet disruption times to a duration commensurate with the failure detection times, two mechanisms may be required:

a. A mechanism for the router(s) adjacent to the failure to rapidly invoke a repair path, which is unaffected by any subsequent convergence.
b. In topologies that are susceptible to micro-loops, a micro-loop control mechanism may be required [RFC5715].

Performing the first task without the second may result in the repair path being starved of traffic and hence being redundant. Performing the second without the first will result in traffic being discarded by the router(s) adjacent to the failure.

Repair paths may always be used in isolation where the failure is short-lived. In this case, the repair paths can be kept in place until the failure is repaired, therefore there is no need to advertise the failure to other routers. Similarly, micro-loop avoidance may be used in isolation to prevent loops arising from pre-planned management action. In which case the link or node being shut down can remain in service for a short time after its removal has been announced into the network, and hence it can function as its own "repair path".

Note that micro-loops may also occur when a link or node is restored to service, and thus a micro-loop avoidance mechanism may be required for both link up and link down cases.

I. Mechanisms for IP Fast-Reroute

The set of mechanisms required for an effective solution to the problem can be broken down into the sub-problems described in this section.

Mechanisms for Fast Failure Detection

It is critical that the failure detection time is minimized. A number of well-documented approaches are possible, such as:.

1. Physical detection; for example, loss of light.
2. Protocol detection that is routing protocol independent; for example, the Bidirectional Failure Detection protocol [BFD].
3. Routing protocol detection; for example, use of "fast Hellos".

When configuring packet-based failure detection mechanisms it is important that consideration be given to the likelihood and consequences of false indications of failure. The incidence of false indication of failure may be minimized by appropriately prioritizing the transmission, reception, and processing of the packets used to detect link or node failure. Note that this is not an issue that is specific to IPFRR.

Mechanisms for Repair Paths

Once a failure has been detected by one of the above mechanisms, traffic that previously traversed the failure is transmitted over one or more repair paths. The design of the repair paths should be such that they can be pre-calculated in anticipation of each local failure and made available for invocation with minimal delay. There are three basic categories of repair paths:

1. Equal cost multi-paths (ECMP). Where such paths exist, and one or more of the alternate paths do not traverse the failure, they may trivially be used as repair paths.
2. Loop-free alternate paths. Such a path exists when a direct neighbour of the router adjacent to the failure has a path to the destination that can be guaranteed not to traverse the failure.
3. Multi-hop repair paths. When there is no feasible loop-free alternate path it may still be possible to locate a router, which is more than one hop away from the router adjacent to thefailure, from which traffic will be forwarded to the destination

without traversing the failure.

ECMP and loop-free alternate paths (as described in [RFC5286]) offer the simplest repair paths and would normally be used when they are available. It is anticipated that around 80% of failures (see Section 5.2.2) can be repaired using these basic methods alone.

Multi-hop repair paths are more complex, both in the computations

required to determine their existence, and in the mechanisms required

to invoke them. They can be further classified as:

a. Mechanisms where one or more alternate FIBs are pre-computed in all routers, and the repaired packet is instructed to be forwarded using a "repair FIB" by some method of per-packet signalling such as detecting a "U-turn" [UTURN], [FIFR] or by marking the packet [SIMULA].

b. Mechanisms functionally equivalent to a loose source route that is invoked using the normal FIB. These include tunnels [TUNNELS], alternative shortest paths [ALT-SP], and label-based mechanisms.

c. Mechanisms employing special addresses or labels that are installed in the FIBs of all routers with routes pre-computed to avoid certain components of the network.

III. Background on P-Cycles

In the arena of WDM or Sonet networking, -cycles are an exciting recent advance because they promise the best properties from each of the basic prior alternatives for restoration: ring and mesh. These are, specifically, the rapid restoration speed of rings and the high capacity efficiency of mesh. Obviously this is an important claim. This section is therefore devoted to substantiating that motivating aspect of the present work, and to provide background about the basic -cycle concept, before going on to consider -cycles in the IP layer. This section can be supplemented for interested readers by references [3], [4], [23]–[25]. Reference [3] is the basic report of our first results with -cycles where we found the total spare capacity required for 100% restoration in five test networks to be within 0 to 9% in excess of an optimal span-restorable mesh, while the real-time restoration switching remained BLSR-like at only two nodes. Reference [4]

describes a distributed autonomous protocol through which a network can self-organize a near optimum set of -cycles within itself, as an interesting alternative to centralized control of -cycle configuration (which remains an option, of course). Reference [23] is a more theoretical analysis substantiating the prior experimental findings of such high capacity efficiency and also proving that -cycles are as efficient as any class of preconfigured spare capacity structure that can exist for restoration. Conference papers [24], [25] overlap somewhat with the present journal paper but [25] also includes material on a nodal capacity-slice device forP – cycle based WDM networking, not published elsewhere, that offers an ADM-like alternative to optical cross-connects for implementation of a WDM -cycle based network. We now give a brief overview on rings, mesh, and -cycles to set the stage for the rest of the work. Ring-based survivability involves the use of bidirectional line switched rings (BLSRs) or unidirectional path-switched rings (UPSRs) as self-protecting transmission systems overlaid on the network topology. Operation and planning of UPSR and BLSR based transpor, and their more recent optical path protection ring (OPPR) and optical shared protection ring (OSPR) variants in a WDM context, is already well covered in the literature. The important point is that rings use a simple switching mechanism which permits restoration in about 50–60 ms, although by their nature they require at least 100% redundancy. In particular, the BLSR uses a working to spare loop-back switching mechanism at the two nodes adjacent to a failure, and this is essentially the identical switching mechanism that -cycles (forWDMor Sonet) employ. In conventional multiring network designs, however, where the working fiber or channel groups themselves are not fully utilizable, effective spare-to-working capacity ratios (the capacity redundancy) can be 200–300%. Thus, rings are fast but not intrinsically capacity-efficient. Mesh-based survivability is more capacity-efficient because each unit of spare capacity is reusable in many ways, across many different failures. Signals that traverse a failed span are rerouted through many diverse paths which, when considered in total, require smaller amounts of spare capacity to realize than are present in ring-based networks. Performance very close to idealized maximum-flow routing efficiency can be achieved. Mesh restoration has traditionally been based on cross-connect systems embedded in a mesh-like set of point-to-point transmission systems, under either centralized or distributed control. Because of this, and because of the more general nature of solving a discrete capacitated multiple-path rerouting problem, restoration is not as fast as with rings but permits a major reduction in the capacity required to serve the same set of demands. While ring-based networks always require 100% or more redundancy, a span-restorable mesh network may be as little as 50% redundant, depending on network topology and demand pattern [7], [8], [11].

Thus, each of these long-standing alternatives has strengths and weaknesses. Mesh networks tend to be economic in long

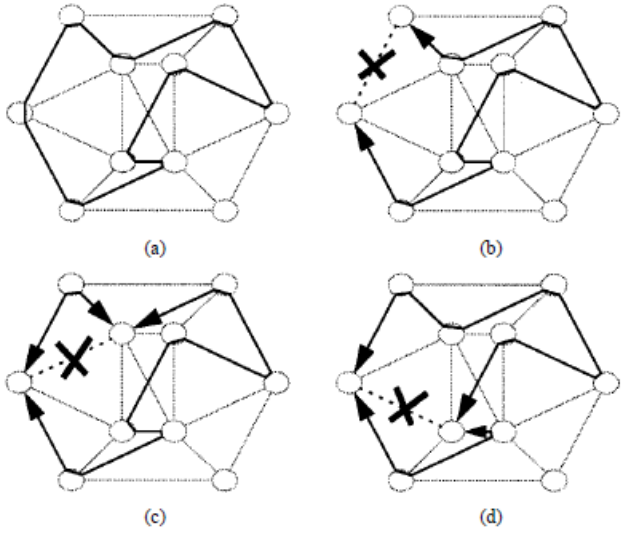

Fig. 1. Use of p-cycles in restoration (from [3]).

haul architectures where capacity efficiency correlates more directly to cost savings. Rings tend to be more cost-efficient in metro areas where cost is dominated by terminal equipment, not distance-related transmission cost. To date, the choice between a ring or mesh-based network has been essentially black or white, i.e., a one-or-the-other proposition. Nothing has previously emerged that can offer the best advantages of both of these extremes. This is the significance of -cycles as first proposed in [3]. They combine the speed of rings with the efficiency of mesh-based networking. An appreciation of this is essential before continuing to IP -cycles.

The method of -cycles for Sonet or WDM is based on the formation of closed paths (elementary cycles in graph theoretic terms), called *-cycles*, in the spare capacity of a mesh-restorable network. They are formed in advance of any failure, out of the previously unconnected spare capacity units of a restorable network. Despite similarity to rings—both use a cycle on the network graph for their topology— -cycles are unlike BLSR/OSPR, UPSR/OPPR (or FDDI) logical rings in that they protect both *on-cycle* and *straddling* failures (to be explained). Fig. 1(a) shows an example of a *-cycle*. In Fig. 1(b), a span on the cycle breaks and the surviving arc of the cycle is used for restoration. This is functionally like a unit-capacity BLSR. In Fig. 1(c) and (d), however, the same -cycle is accessed to support restoration of working paths that are *straddling* the cycle. In fact, cases Fig. 1(c) and (d) are the more advantageous circumstances in general because *two* restoration paths are available from each -cycle for such failures. In contrast, either type of conventional Sonet or WDM ring provides at most one restoration path per unit of ring protection capacity. Rings also protect only against failures on the spans of the same ring, not on "straddling" spans. This makes a significant difference to the network restoration coverage provided by the same investment in spare capacity in a ring as opposed to in a -cycle. For example, further examination of the single -cycle in Fig. 1 shows that it can provide restoration path(s) to 19 potential span failures (ten straddling relationships, nine on-cycle relationships), while as a ring, protectionis available only for the nine spans on the cycle. But, in addition, the -cycle provides two restoration paths for each of the ten spans that are in a *straddling* relationship. Thus, spare capacity on a -cycle is more widely accessible, i.e., more highly shared for restoration than in a BLSR or UPSR. Although it is not initially obvious, under the appropriate design optimization, this fact allows -cycle based networks to be essentially as capacity- efficient as mesh networks. This was verified in [3] where fully restorable -cycle capacity plans were generated and compared to conventional mesh restoration for five test networks. The worst test case required 9% additional spare capacity while the remaining cases required 0 to 3%. Reference [3] details the test case networks and the mixed integer programming formulation under which these results were obtained. Although -cycles seem, initially, to embody only one small difference relative to today's well-known ring systems (the aspect of protecting straddling failures), there are many consequences from this difference when fully worked through. p-cycles (if based on cross-connects) can be formed from unit capacity channels of the point-to-point OC-n or DWDM systems present, whereas rings commit a whole OC-n module of working and spare capacity to the same cycle. Rings also have a structural association between the working demands which they protect and the protection bandwidth in the same ring, while -cycles are formed only within the spare capacity layer of the network, leaving the working paths to be routed freely on shortest paths, or any other route desired. In other words, the working demands may be provisioned freely as growth arises, as if in a failure-free point-to-point network; the -cycles formed in the sparing layer adapt to suit the working path layer. A deployed -cycle design may also be easily modified by the cross-connects that form it, whereas Sonet ring placements are essentially permanent structural Commitments of both working and spare capacity, to which the routing of new working paths must conform. Finally, the implication of protecting straddling failures is that a -cycle

spare capacity design takes little or no more capacity than a corresponding span-restorable mesh network [3]. Generally this will be substantially less than 100% capacity redundancy. This property, plus the fact that the switching operations for restoration with -cycles is essentially just that of a BLSR, is the reason we say that -cycles can offer "the speed of rings, with the efficiency of mesh." The reason they are as fast as rings is that there are only two traffic-substituting connections to be made for any working path failure to be restored. Moreover, the two end-nodes that perform the switching only do a transmit signal bridging and receive direction transfer operation that is essentially BLSR-like in nature, and they know in advance exactly which working-to spare switching functions will be needed for any given failure.

III. E-CYCLE: ENHANCED PROTECTION CYCLE FOR

ROUTING PROTECTION: In this section, we present the design of *e-cycle*, a solution using enhanced protection cycles for routing protection. We first present the overview of *e-cycle* and then propose different detailed algorithms to build *e-cycle* for different routing protocols.

A. Overview of e-cycle

Different from traditional auto-discovery protection solutions which introduce high deployment complexity to routing protocols [5], [15], [12], *e-cycle* provides efficient preconfigured routing paths to realize fast rerouting. Similar to *pcycle*[31], *e-cycle* leverages virtual cycles to construct rerouting paths and uses different identifiers called *e-cycle IDs* (see discussion below) to uniquely identify these virtual cycles, thus provides protection for all nodes and links. The main difference is that, since every router has routes to destinations, *e-cycle* does not detour packets along an entire virtual cycle as in *p-cycle*, but seeks to find an earlier decapsulation point after which packets are again forwarded along normal routes. To achieve this, *e-cycle* introduces two components, namely, *protection initiators (PIs)* and *protection terminators (PTs)*. Protection initiators (PIs) are routers that detect failures and then activate protection paths to forward packets, and protection terminators (PTs) are routers that terminate protection paths and continue normal packet forwarding. If a router detects a failure, then it will activate itself to become a PI, and will select a corresponding PT. We will discuss the PT selection in the following discussion.The main idea of our e-cycle approach is that when an ecycle is constructed, we select a PT for every PI in the cycle and packets are only forwarded along the partial cycle between the PI and PT. When a PI detects a failure, it starts to forward affected packets along the e-cycle towards its corresponding PT. Since we want to introduce as little overhead as possible to realize routing protection, we propose label e-cycle ID based direct forwarding. An e-cycle ID specifies the unique identifier of the e-cycle (i.e., virtual cycle) that is used for rerouting. It can be manually configured in routers that have deployed ecycle, or distributed through an automatic mechanism such as Label Distribution Protocol (LDP) as in Not-via [15] 1. To specify the correct e-cycle ID (and hence the e-cycle) in packets to be forwarded, each PI can simply encapsulate the packets with a new IP header using IP encapsulation (e.g., L2TP [33]) to keep backward compatibility, and the new IP header will contain an e-cycle ID field. In addition, we also include a hop count field in the new IP header. The hop count field specifies the hop count between PI and PT. It is used to indicate the lifetime of a packet in the e-cycle, and will be decremented by one when the packet is forwarded by a router. If the hop count equals to zero, then the packet will be removed from the e-cycle by the PT and the original packet will be forwarded along a normal route to its destination 2. Figure 3 illustrates how e-cycle addresses the same failure as in Figure 2. Assuming R5 as a PI, we can choose R3 as the PT for R5 because the route to R1 in R3 will not pass though R5. R3 removes the e-cycle header and forwards it normally, and the length of the rerouting path in e-cycle is only 2. Thus, we can achieve an effective lightweight protection for intra-domain routing and further provide connectivity between iBGP speakers. To provide

protection for eBGP, it is important to note that the e-cycle approach does not require all nodes in the cycle to have deployed e-cycle and to be configured with the same e-cycle ID. As shown in Figure 3, we assume that AS2 and AS3 are two provider ASes of AS1 and a virtual cycle (R1-R3-R6-R7-R9-_-R8) (_ denotes a sequence of traversed routers in which we do not need to configure ecycle for eBGP protection) has been constructed. When link R1-R8 fails, R1 will detour packets along R3-R6-R7 to R9 and R9 definitely has routes to destinations. Note that, in Figure 3, both AS 2 and AS 3 are provider ASes of AS 1, and then they can provide transit service for AS1. For each destination, AS 2 and AS 3 can help AS 1 deliver packets to it. For any one link failure, e.g., link AS1-AS3 fails, AS 2 can help AS 1 deliver packets under failure with eBGP link AS2-AS3. If AS 1 has a routing reconvergence process after the failure, it will eventually select the route with the eBGP link AS2-AS3 as the best one. In this sense, it will not violate the routing policies. For the reverse traffic from AS 3 to AS 1, the situation is similar. For the failure of link AS1- AS3, if AS 2 and AS 3 deploy an e-cycle, AS 3 can deliver the packets to AS 1 with the help AS 2. However, if AS 2 and AS 3 do not have negotiation between themselves, AS 3 will still deliver packets to AS 1 with IP header. Normally, the IP addresses of eBGP link in R3 and R8 are managed by the same ISP (/AS), AS 1 or AS 2, and the addresses are known by both two ASes. Thus, we can safely encapsulate the packets to AS 1 with R8's IP address such that the encapsulated packets will be delivered to AS 2 according to the routing tables. After the packets reach R8, R8 will decapsulate the packets and send the packets to R3 in AS 1 eventually. Note that, the agreement between AS 2 and AS 1 may be required to protect the traffic from AS 1 to AS 3 over link AS1-AS3. However, AS 3 does not need to build agreements with AS 2 to protect traffic from AS 3 to AS 1 because the destination of the traffic to AS 1 will be encapsulated with R8's address which connects AS 2. AS 2 can directly forward the packets to AS 1, while not requiring decapsulating them for AS 1. The network configurations conform to the normal network operation practice. Similarly, if we enable routing re-convergence after link failure, the packets to AS 1 will finally get to AS 2 and AS 2 delivers the packets to final destinations. The difference between *e-cycle* and traditional routing reconvergence scheme is that *e-cycle* Provides fast rerouting to deliver packets without routing reconvergence involved. There are several types of failures that *e-cycle* must handle. For link failures, the failed link may or may not lie on the preconfigured *e-cycle*, and for node failures, the adjacent router may or may not lie on the same *e-cycle* as the failed one. Thus, our *e-cycle* solution should still be able to handle all these conditions by detouring packets to the corresponding PT as long as an *e-cycle* is pre-configured on a PI. Thus, we expect that *e-cycle* provides much better efficiency by realizing a unified protection for both node and link failures. Given that intra- and inter-domain routing protocols have different forwarding features, we should have different *ecycle* construction methods for the protocols. In the following subsections, we will discuss two main issues: (i) how to construct *e-cycles*, and (ii) how to select the PT, in order toprotect routing failures for different types of routing protocols.

B. Intra-Domain Routing/iBGP Link Protection

We first describe how to provide link protection to intra-domain routing using e-cycle. Since iBGP relies on intra-domain routing, if we can guarantee link protection in intra-domain routing, then iBGP link failures can be eliminated. Note that the protection for an iBGP node is more complex because the node may be the only one egress point within an AS, and intra-domain protection can't successfully provide failure recovery. We will discuss this in Section III-D. So next we only discuss e-cycle construction for intra-domain routing and iBGP link protection. We assume that intra-domain routing uses shortest path routing (e.g., OSPF and ISIS), and each router has formed a shortest path tree (SPT) that specifies all routing paths to other nodes within the domain. Also, given that virtual cycle construction in IP networks is well studied in the literature [31], [30], we leverage the same construction

algorithm as p-cycle to construct virtual cycles.In the following discussion, we focus on the PT selection in the virtual cycles that have been constructed. Algorithm 1 shows the intra-domain e-cycle construction algorithm. It returns a set C, in which each member c is a virtual cycle composed of the set of routers Vc in the cycle and the corresponding set of PTs Sc. First, we construct candidate virtual cycles using existing algorithms [31], [30](step 1). Then for each cycle c, we choose a PT for each router Rci On c (step 4-19). Note that c is uni-directional, and the nodes in Vc are ordered (in a cycle) as [Rc1, . Rci-1, Rci, Rci+1, … Rcm] such that when we traverse the cycle starting from Rci, Rc i+1 is the next node to be encountered and Rci−1 is the last one. In the shortest path tree (SPT) rooted at Rci, if SPT Desc(Rci) returns the descendants of the subtree under the failed link (or failed node), then we try to find a router Rcx in the cycle c, such that the shortest path from Rcx to any router Ry inSPT Desc(Rci) does not pass Rci. That is, if Cruises Rcx to detour the failed link and forward packets to its descendent routers, then Rcx will never send the packets back to Rci since the cost from Rcx to Ry should be less than that from Rc i toRy. If such a router is found, then we can directly set Rcx as the PT of Rci in the cycle c (step 5-16). Otherwise, the router Rc i+1 next to Rci along the cycle c will be chosen as the PT (step 17-19). For an e-cycle, a PI only needs one PT. In Algorithm 1, we will choose PT for PI if PT can be used to detour the failure and forward traffic to all destinations. In the worst case scenario, PT is the opposite to PI in the assumed failed link. We now explain Algorithm 1 with an example. Figure 4 shows an intra-domain topology where the link weights are all set to 10, except that the weight of R5-R3 is 11. According to the link weights, all shortest paths root at different nodes are determined Once a PT is chosen, we need to distribute the alternate forwarding entries to the routers on this cycle for identifying the *e-cycle* ID. Figure 4 shows that we construct two *e-cycles* for eight routers in the AS. If any router detects a failure, then it can launch the protection with a specific PT in the cycle. For example, as shown in Figure 4, R5 acting as the PI activates the protection path to R3 once it detects the failure on R1-R5, and R2 activates the protection path to R7 once it detects the failure on R2-R6. In this way, traffic for R6 will go through R5, R4, R3, R2, R3 and R7, and finally beforwarded to R6 using normal route by R7.

Algorithm 1 Intra-domain *E-cycle* Construction

//SPT_Desc(R): the descendants of the subtree under the failed link (or failed node) in the SPT rooted at R;
//SPT_traversed(R_x, R_y): the set of routers along the shortest path from R_x to R_y.
Input: intra-domain topology;
Output: $C = \{c | c = (V_c, S_c)\}$;

1: construct *virtual cycles* $C=\{c|c = (V_c, \emptyset)\}$;
2: **for** each $c \in C$ **do**
3: **for** each $R_i^c \in V_c$ **do**
4: flag = **true**;
5: **for** (R_x^c in [R_{i+1}^c, R_{i+2}^c, ..., R_m^c, R_1^c..., R_{i-1}^c]) **do**
6: **for** each R_y in $SPT\_Desc(R_i^c)$ **do**
7: **if** ($R_i^c \in SPT\_traversed(R_x^c, R_y)$) **then**
8: flag = **false**;
9: **break**;
10: **end if**
11: **end for**
12: **if** (flag == **true**) **then**
13: $update\_PT(c, R_i^c, R_x^c)$;
14: **break**;
15: **end if**
16: **end for**
17: **if** (flag == **false**) **then**
18: $update\_PT(c, R_i^c, R_{i+1}^c)$;
19: **end if**
20: **end for**
21: **end for**

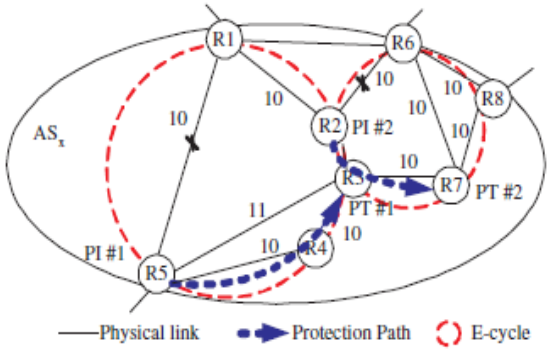

Fig. 4. *E-cycles* in intra-domain routing protection.

V.Performance Analysis

In this section, we evaluate the performance of our *e-cycle* approach by simulations. We firstly describe simulation setup in Section IV-A and then present the simulation results in Section IV-B.

A. Simulation Setup

To evaluate our proposed solution, we implement asimulator that is able to simulate both intra- and inter-domain routing protocols [38]. In particular, the simulator considers BGP policy configurations, so that it can accurately evaluate the per formance of eBGP protection. Our simulator simulates how a router would protect all end-to-end routing paths with different solutions including traditional IP-FRR [11] Lightweight Not-Via [27], BGP-FRR [5], and R-BGP [12]. For each link in an end-to-end routing path, if no protection path is found, the simulator determines that the protection solution fails and can not provide protection for this failure. Since the p-cycle adoption proposed by Stamatelakis et al. [31] is similar to the original p-cycle solution [29] and they both can not be directly applied to IP networks, we only evaluate the performance of the original p-cycle. Normally, each node deployed with routing protection solutions activates a protection path to reroute the packets within several hundred microseconds after it detects the failure. However, the node will spend much more time in computing the rerouting path if the routing protection solution is not deployed in the node.

B. Experimental Results

Experiment 1 (Packet loss): We randomly choose ten different times in five different days to generate 100 ping packets and compare the average packet loss rate of traffic. Figure 15 shows the packet loss rate of different flows with different routing paths. In Figure 15(a), we measure the loss rate of small packets with different paths. It shows that the protection path reduces the packet losses of the normal path. While the activation of the protection path (i.e., path switching) may trigger some packet loss, the loss rate remains lower than that in the normal path. Overall, we observe that the use of the protection path can reduce the packet loss rate resulting from the normal path (which experiences congestion in our experiments). We will compare the performance in the normal path and the protection path under different situations with different experiments.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a unified protection solution called *e-cycle* for intra- and inter-domain routing to efficiently recover from routing failures. Specifically, *e-cycle* constructs protection paths and provides node and link protection. Simulation results show that our proposed solution achieves 100% failure coverage in both intra- and inter-domain routing. Moreover, we partially deployed our solution in operational networks to demonstrate the practicality of our solution. We find that our solution will not introduce much overhead to packet forwarding.

VI.References

[1] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed Internet routing convergence," IEEE/ACM Trans. Networking, vol. 9, no. 3, pp.293-306, 2001.

[2] Y. Afek, A. Bremler-Barr, and S. Schwarz, "Improved BGP convergence via ghost flushing," IEEE J. Sel. Areas Commun., vol. 22, no. 10, pp.1933-1948, 2004.

[3] D. Pei, M. Azuma, D. Massey, and L. Zhang, "BGP-RCN: improving BGP convergence through root cause notification," Computer Network,vol. 48, no. 2, pp. 175-194, 2005.

[4] O. Bonaventure, C. Filsfils, and P. Francois, "Achieving sub-50 mil-liseconds recovery upon BGP peering link failures," IEEE/ACM Trans. Networking, vol. 15, no. 5, pp. 1123-1135, 2007

.[5] M. Shand and S. Bryant, "IP fast reroute framework," RFC5714, Jan.2010.

[6] D. Katz and D. Ward, "Bidirectional forwarding detection," RFC 5880,June 2010.

[7] P. Mérindol, V. Van den Schrieck, B. Donnet, O. Bonaventure, and J.-J. Pansiot, "Quantifying ASES

multiconnectivity using multicast information," in Proc. 2009 IMC, pp. 370-376.

[8] B. Wu, K. L. Yeung, and P. H. Ho, "ILP formulations for p-cycle design without candidate cycle enumeration," IEEE/ACM Trans. Networking,vol. 18, no. 1, pp. 284-295, 2010.

[9] D. Stamatelakis and W. Grover, "P-cycles: IP layer restoration and network planning based on virtual protection cycles," IEEE J. Sel. Areas Commun., vol. 18, no. 10, Oct. 2000.

[10] M. Gjoka, V. Ram, and X. Yang, "Evaluation of IP Fast Reroute Proposals", Proc. Of COMSWARE, Bangalore, India, Jan 2007.

[11] A. Li, P. Francois, and X. Yang, "On improving the efficiency and manageability of NotVia", Proc. of ACM CoNEXT, Dec 2007.

[12] P. Francois, "Improving the Convergence of IP Routing Protocols", PhD thesis, Universit´e catholique de Louvain, Oct 2007.

[13] S. Bryant, M. Shand, and S. Previdi, "IP Fast Reroute Using Notvia Addresses", Internet Draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-04.txt,July 2009.

[14] G. Enyedi, G. R´etv´ari, P. Szil´agyi, and A. Cs´asz´ar, "IP Fast ReRoute:Lightweight Not-Via without Additional Addresses", IEEE Infocom, Apr 2009.

[15] K. Ho, N. Wang, G. Pavlou, and C. Botsiaris, "Optimizing post-failure network performance for IP Fast ReRoute using tunnels", Proceedings of QShine, Hong Kong, July 2008.

[16] S. Bryant, C. Filsfils, S. Previdi, and M. Shand. IP Fast Reroute using Tunnels. Internet draft, draft-bryant-ipfrr-tunnels-01.txt,work in progress, Oct 2004.

[17] A. Atlas. U-turn alternates for IP/LDP Local Protection. Internet draft, draft-atlas-ip-local-protect-uturn-00.txt, work in progress, November 2004.

[18] S. Bryant and M. Shand. IP Fast Reroute using Notvia Addresses. Internet draft, draft-bryant-shand-ipfrr-notvia-addresses-00.txt, work in progress, March 2005.

[19] Q. Li, M. Xu, L. Pan, and Y. Cui. A study of path protection in selfhealing routing. In Proceeding of IFIP Networking, pages 554–561,2008.

[20] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, and C. Diot. Characterization of failures in an IP backbone. In Proceeding of the IEEE INFOCOM, pages 2307– 2317, 2004.

[21] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," Feb. 2002.

[22] A. Nucci, B. Schroeder, S. Bhattacharyya, N. Taft, and C. Diot, "IGP link weight assignment for transient link failures," in ITC18, Sept. 2003.

[23] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "Feasibility of IP restoration in a tier-1 backbone," IEEE Network Magazine, Special Issue on Protection, Restoration and Disaster Recovery, 2004.

[24] D. Applegate and E. Cohen. Making intra-domain routing robust to changing and uncertain trafc demands: Understanding fundamental tradeoffs. In Proceedings of ACM SIGCOMM '03, Karlsruhe,Germany, Aug. 2003.

[25] M. Conforti, A. Galluccio, and G. Proietti. Edge-connectivity augmentation and network matrices. In J. Hromkovic, M. Nagl, and B. Westfechtel, editors, WG, volume 3353 of Lecture Notes in Computer Science, pages 355.364. Springer, 2004.

[26] B. Fortz and M. Thorup. Robust optimization of OSPF/IS-IS weights. In Proceedings of INOC, Oct. 2003.

