This is a postprint version of the following published document:

L. Roelens, Ó. González de Dios, I. de Miguel, E. Echeverry and R. J. Durán Barroso, "Performance Evaluation of TI-LFA in Traffic-Engineered Segment Routing-Based Networks," *2023 19th International Conference on the Design of Reliable Communication Networks (DRCN)*, Vilanova i la Geltru, Spain, 2023, pp. 1-8, https://doi.org/10.1109/DRCN57075.2023.10108280

© 2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Performance Evaluation of TI-LFA in Traffic-Engineered Segment Routing-Based Networks

Liesbeth Roelens
Telefónica I+D
Madrid, Spain
0000-0002-9756-2469

Óscar González de Dios
Telefónica I+D
Madrid, Spain
0000-0002-1898-0807

Ignacio de Miguel
Universidad de Valladolid
Valladolid, Spain
0000-0002-1084-1159

Edward Echeverry
Telefónica SA
Madrid, Spain

Ramón J. Durán Barroso
Universidad de Valladolid
Valladolid, Spain
0000-0003-1423-1646

***Abstract*—Link failures have a significant negative impact on the availability of a network and should therefore be resolved as soon as possible. Because of the slow convergence time of routing protocols upon detection of a link failure, several IP Fast ReRoute (FRR) mechanisms have been developed to overcome this problem. Recently, segment routing, which is a flexible and scalable way of doing source routing, enabled a new FRR mechanism called Topology Independent Loop-Free Alternate (TI-LFA). As the name suggests, the key feature of TI-LFA is that it guarantees a loop-free detour against any link failure in any network topology. However, typically fast responses to failures only aim to restore the loop-free connection between the affected routers and do not consider the resulting delay or impact on network congestion. This paper presents an initial study on the selected TI-LFA backup paths and their effect on the overall network performance. By means of simulation, we evaluate how efficient TI-LFA reroutes traffic for a number of traffic engineering approaches. Our results quantify the impact of different traffic engineering approaches and network loads on the performance of TI-LFA. This suggests potential directions for improving the effectiveness of TI-LFA protection in segment routing.**

***Index Terms*—Segment Routing, Link Failures, IP Fast-ReRoute, TI-LFA, Discrete Event Simulation**

This work is part of IoTalentum project that has received funding from the EU H2020 research and innovation programme under the MSCA grant agreement No 953442. It is also supported by H2020-ICT-52-2020 TeraFlow project (grant 101015857) and the Spanish Ministry of Science of Innovation and the State Research Agency (Grant PID2020-112675RB-C42 funded by MCIN/AEI/10.13039/501100011033)

978-1-6654-7598-3/23/$31.00 ©2023 IEEE

I. Introduction

The widespread adoption of high-bandwidth, low-latency applications in communication networks is constantly pushing network operators towards more stringent Service Level Agreements (SLA) that guarantee peak performance, which is often defined as delivering services to users 99,999% of the time. Unfortunately, network failures are inevitable and occur for a variety of reasons, including short-term router interface faults, router crashes and reboots, or worst-case even long-term fiber cuts [1]. These failures pose a threat to the SLAs and require an immediate reaction to secure seamless connectivity. In contrast, the global network convergence after detecting a failure is relatively slow, ranging from seconds to several minutes [2], which is insufficient to meet SLAs with strict guarantees. The slow convergence time is due to the delay in propagating the fault message to a central Software-Defined Networking (SDN) controller and the new path computations. Fast ReRoute (FRR) mechanisms have been designed to quickly handle unexpected failures by acting *locally* in the *data* plane [3]. These mechanisms aim to minimize downtime in the network by pre-installing static backup routes. In this way, a node that detects a failure is 'prepared' and can forward the packets instantaneously. Various FRR approaches have been developed for this purpose [4].

In this work, we consider Topology Independent Loop-Free Alternate (TI-LFA), an FRR mechanism enabled by segment routing (SR) [5], [6]. SR is a source-based routing architecture that originated to sim-

plify scalability issues in Multiprotocol Label Switching (MPLS)-based Traffic Engineering (TE) methods [7]. SR accomplishes traffic steering by decomposing a forwarding path into an ordered list of segments (known as an SR policy), which is pushed as a *label stack* in the packet header of the ingress node. By specifying intermediate waypoints in the label stack, path diversity can extend beyond shortest path routing. While this is beneficial for traffic engineering, FRR methods can equally benefit from these intermediate waypoints. This led to the emergence of TI-LFA, which is the standard FRR method to protect links and nodes in SR. The basic principle of TI-LFA is to pre-compute the post-convergence path from the Point of Local Repair (PLR), i.e. the node detecting the failure, to the destination. TI-LFA can now simply deploy SR to express this post-convergence path via a stack of labels, or a *repair list*, and store it locally at the PLR. The procedure is then repeated for all destinations. Whereas previous FRR methods were often topology-dependent and resulted in loops, TI-LFA can protect all links and all nodes in all network topologies. Nonetheless, despite having a *working* FRR method for SR networks, the *optimal* FRR path should comprise more criteria, the two most important being the resulting traffic load and latency. Defining the optimal FRR path in terms of load and latency is not a straightforward task [8]. When traffic is rerouted using a longer backup path, it has more latency and also consumes more bandwidth resources. On the other hand, in heavily loaded networks an alternative, longer backup path may be useful when the default backup path is congested. Keeping in mind the progressively stricter SLA requirements of many IoT, voice, and video applications, we believe these aspects should not be overlooked when studying FRR mechanisms such as TI-LFA. The main challenge, however, is that FRR methods are locally pre-computed. More precisely, because there is no time to communicate dynamic network state information, these methods are unaware of whether they take unnecessary detours or interfere with other traffic, resulting in costly packet loss. For this reason, it is a non-trivial problem how TI-LFA can provide optimal backup paths while not exploiting the full connectivity of the underlying network.

This paper presents an initial study and simulation of the static backup paths provided via TI-LFA using SR-TE methods in a small network topology, focusing on the resulting length of the backup paths (latency) and network load. As Section 9.1 of RFC 9256 [7] acknowledges "*Since TI-LFA protection is based on IGP computation, there are cases where the path used during the fast-reroute time window may not meet the exact constraints of the SR Policy*". Thus, our study aims to address the following observations:

1) Although TI-LFA is able to provide 100% topological connectivity, this does not necessarily result in the most efficient backup path, since the local PLR does not have a global picture of the network upon failure detection.
2) In line with the previous observation, inefficient backup paths inherently lead to additional traffic load in the network. Accordingly, we analyse to what extent TI-LFA backup paths negatively affect latency and network congestion, and whether there is potential for optimisation.

The remainder of the paper is structured as follows. In Section II, we discuss the network model, the SR-TE mechanisms that we consider and a motivation for testing the performance of TI-LFA in this environment. In Section III, we describe in more detail how the simulation is performed. In Section IV, we evaluate the network performance, both for the scenario with and without link failures, and discuss the performance of TI-LFA fast restoration in the latter case. Section V concludes the paper and provides some directions for future work.

II. Network Model and Problem Description

The network topology is modelled as a graph consisting of N nodes connected via undirected point-to-point links. We assume all network nodes are SR-enabled. Then, SR policies (tunnels, from now on) that steer traffic flows in the network are established following two steps.

In the first step, the path to be followed from source to destination is determined by a centralised SDN controller. For this, the Shortest Path (SP) can be used. However, we can also leverage the flexibility provided by segment routing to choose a different path instead. In this work, besides the SP we also consider the Shortest Available Path (SAP). This SAP is computed by dynamically filtering the links in the network that provide sufficient capacity to support the bandwidth of a new traffic demand. The shortest path is then determined based on this residual graph.

The second step is to encode the selected path. In segment routing, a packet is encoded as a stack of labels or Segment Identifiers (SIDs) inserted in the packet header by the ingress node. Each label corresponds to a SID, which can be a node or adjacency (link) in the scope of our model. When the top label on the stack is a node SID, it does not need to be directly

adjacent to the current node, as SR relies on shortest path routing in between. It is worth noting that we do not consider traffic splitting when multiple shortest paths between two nodes are available. In case of an adjacency SID, it needs to be directly adjacent to the current node. We apply two approaches. The first one is a naive approach in which every node's SID along the path is pushed to the segment stack. We denote this approach as node encoding. However, more intelligent algorithms can express the desired path whilst keeping the required number of labels to a minimum, like the Segment Routing – Label Encoding Algorithm (SR-LEA) [9]. SR-LEA uses both global node SIDs and local adjacency (link) SIDs to enforce any path with minimal labels.

Our aim is to compare the performance of the TI-LFA Fast ReRoute mechanism when various SR-TE approaches are applied in the network, namely, the use of SP vs SAP algorithms, as well as the use of node encoding vs SR-LEA.

To make this more concrete, we show in Fig. 1 an example of how TI-LFA guarantees connectivity after a link failure, but fails to take the most efficient path to the destination. In this figure, an SR path is established from source node s to destination node d, forcing traffic to take the upper path by specifying node x as an intermediate label in the segment stack. When s detects an error on its outbound interface to x, TI-LFA instantly pushes its pre-configured repair list on top of the segment list to reroute traffic to the next *segment*, i.e. node x. Now, incoming traffic at s that was supposed to follow the green SR path is rerouted via the red path. We see that although TI-LFA succeeded in rerouting traffic, an optimal repair path would not route traffic beyond node d (to x and then back to d). Sending packets back and forth across the links between x and d causes unnecessary load and delays in the network and should ideally be avoided. As we will show, the methods selected for routing (SP or SAP) and encoding (node encoding on SR-LEA) have an impact on this issue. Therefore, we will compare, by means of simulation, which method yields the best results for TI-LFA.

In particular, we measure the performance of TI-LFA in terms of the following metrics:

- *Success rate*: This metric measures the percentage of flows that can be successfully recovered after a link failure (among the flows affected by that failure). When dealing with a link failure, the backup path may already be in use by other traffic flows, so rerouting may overload links and thus it cannot be really used due to bandwidth unavailability. Ideally the rerouted flows should cause minimal additional load for each link.
- *Path stretch*: For successfully restored flows, this metric reports the increase in the number of hops for the new route compared to the primary route, both in absolute and relative (percentage) terms. Longer backup paths introduce more load and latency in the network and are therefore worth measuring.
- *Node repetitions*: This metric indicates how many nodes are visited more than once in the backup path. Detours as in Fig. 1 generate unnecessary overhead in the network and should ideally never occur. For instance, the value of this metric for the scenario in Fig. 1 would be 3, as three nodes are visited twice by the recovered traffic flow.

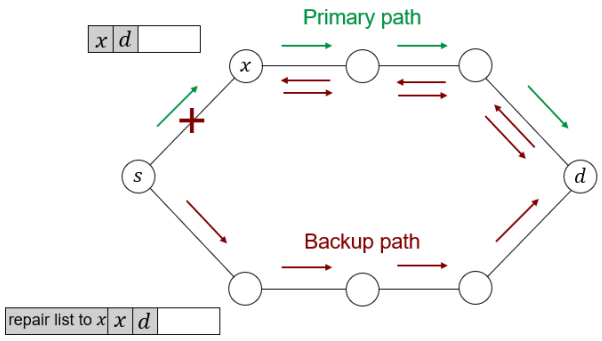


Fig. 1: A graph with 2 multi-hop paths between source s and destination d. Traffic is forced to take the upper path (primary path) by inserting node x in the segment list. Upon detection of a failure, s pushes its TI-LFA repair list on top of the segment list, resulting in node repetitions along the backup path.

III. Simulation Methodology

We have developed a discrete event-based network simulator (DES) that provides support for SR-TE when traffic requests are dynamically arriving at the network, and for performing fast restoration using TI-LFA. The simulator has been developed using the Python-based software framework SimPy [10]. Essentially, SimPy allows modelling asynchronous tasks through a set of basic operations such as events, processes and resources.

Fig. 2 provides a conceptual representation of the simulator and network model. The key idea behind the simulator is to bring the network to a state where a set of tunnels (for the transmission of traffic flows) is established according to a certain load, and then assess

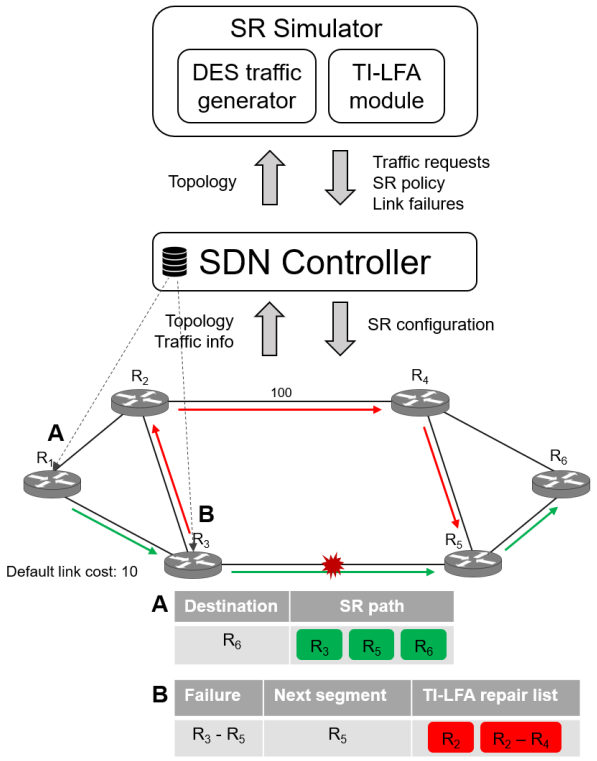


Fig. 2: A conceptual representation of the SR-TE simulator and TI-LFA backup paths.

how many of those traffic flows would be successfully recovered for each potential single link failure if TI-LFA is used. The metrics associated to the recovered tunnels, namely the increase in length and occurrence of node repetitions, are also computed. While keeping the traffic load (the average number of tunnels set up in the network) to a fixed value, the procedure is then repeated at different moments, so that the set of established tunnels is different at each moment, to obtain statistically meaningful results.

To do so, tunnels are dynamically established and released in the network, and periodically (after sufficient changes in the network) a 'snapshot' of the network state at that moment is taken. In such a snapshot, all potential single-link failures that disrupt an existing traffic flow are identified and analysed individually. Iteratively, for every possible link failure, the TI-LFA repair list is pushed on top of the remaining segment list, and we then determine whether the traffic flow can be successfully rerouted to its destination. If true, we save the full new path and retrieve its metrics. On a side note, to encode TI-LFA post-convergence paths as repair lists, the basic algorithm presented in [11] is applied. Also, as this paper focuses on the *fast* restoration of link failures, we do not include later, *slow* convergence effects, in our model. We are only interested in the instantaneous effect of a link failure. In between two snapshots where the "what-if" scenario associated to each single link failure is considered, the simulator continues its usual operation dynamically establishing and releasing tunnels. Note that this dynamic establishment and release of tunnels only serves to ensure that the network evolves to a different state before analysing the impact of failures in that scenario, allowing failures to be tested during different network states during the simulation.

In order to establish and release tunnels, the module 'SR simulator' generates bidirectional traffic flows from a specific source to a destination node as a chronological sequence of discrete events. It also provides the SR-TE policy and the duration for establishing the tunnel in the network. The SDN controller has full knowledge of the network topology and communicates with the nodes to centrally manage the requested traffic flows and steer them through the network according to the provided SR-TE policy. An example of a deployed SR tunnel for a traffic flow originating at R_1 is displayed in Fig. 2. Furthermore, at each node we store pre-computed TI-LFA backup paths for link failures to all possible destinations. The SR configuration of such a backup path, i.e. the repair list, is visualised with an example in R_2. Ultimately, the SR simulator will use all collected data to assess the optimality of the implemented fast failover routes.

We assume that traffic flows (the requests for setting up a tunnel) arrive at the network according to a Poisson process with inter-arrival rate λ. The source and destination nodes for each traffic flow are randomly selected according to a uniform distribution. Each traffic flow requests a fixed capacity, denoted as C_{cxn}. The duration for which an SR tunnel is being established follows an exponential distribution for which we specify a fixed mean T. Then, the normalised traffic load ρ is defined by Equation 1,

$$\rho = \frac{\lambda \cdot T}{N \cdot (N-1)/2} \cdot \frac{C_{\text{cxn}}}{C_{\text{link}}} \tag{1}$$

where N represents the number of nodes in the network and C_{link} the capacity of a link. For simplicity, all links have the same capacity so that the ratio $C_{\text{cxn}}/C_{\text{link}}$ remains constant during the simulation. Ultimately, the load is normalised to take the dimensions of the considered network into account as well. A load $\rho = 1$ can be interpreted as a network where on average one

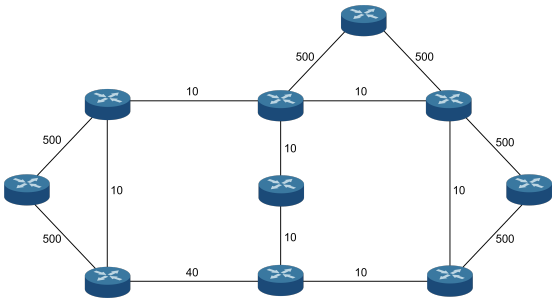


Fig. 3: Network considered in the evaluation. Each link is tagged with its cost. All links have the same capacity.

bidirectional traffic flow is carried between each source-destination pair, consuming the full bandwidth of the links (or similarly, two bidirectional flows consuming half the bandwidth).

IV. Performance Evaluation

A. Simulation Setup

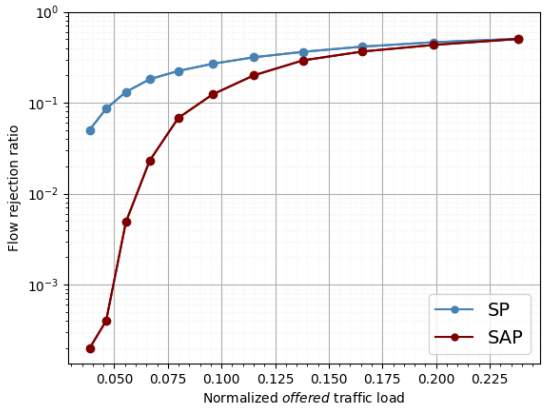
To evaluate the discussed SR-TE methods for path selection and encoding in our simulator, we feed 11 000 tunnel establishment requests (to transport traffic flows) to the network topology depicted in 3, which consists of 10 SR-enabled routers connected with 14 undirected links. To remove possible transient effects at the start of the simulation, the first 1 000 traffic flows are eventually discarded from the evaluation. With the remaining 10 000 traffic flows, we determine the traffic *flow rejection ratio* and *average tunnel length*. To account for different network loads, the simulation is run repeatedly for gradually increasing values of the inter-arrival rate λ in Eq. 1, while the mean duration T of a traffic flow is kept constant and the ratio $C_{\text{cxn}}/C_{\text{link}}$ is fixed at 0,10.

Next, we analyse the scenario which addresses single link failures and fast recovery with TI-LFA. To define the periodic update interval to generate snapshots in the network, we set this value equal to the flow inter-arrival times obtained for different traffic loads. This guarantees the same number of snapshots in every simulation. We obtain the metrics discussed in Section II and average them over all link failures within one snapshot, after which they are averaged a second time over all snapshots during the simulation.

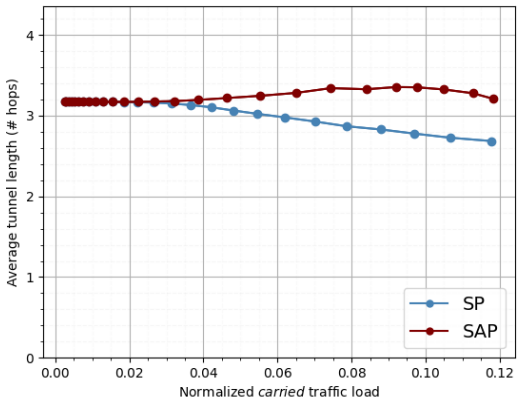
B. Simulation Results

Fig. 4 represents the performance of the discussed SR-TE approaches in the absence of link failures. The results when SR tunnels employ SP are represented in blue, while red is used for SAP. The performance is analysed in terms of flow rejection ratio (Fig. 4a), i.e. the ratio of rejected SR tunnels (due to lack of resources) to requested tunnels, and the average tunnel length (Fig. 4b), expressed in number of hops. Each data point is obtained by averaging the measurements of 10 000 incoming traffic requests. The approach for encoding the path (node encoding or SR-LEA) does not have an impact on these results and is therefore omitted here. The x-axis shows increasing values for the (normalised) traffic load applied during the simulation. Here, a distinction is made between the *offered* and *carried* traffic load. In Fig. 4a, the offered load refers to the requested amount of traffic flows we want to establish in the network, whereas in Fig. 4b, the carried load reflects the actual traffic present in the network (before facing failures). As traffic is homogeneously generated, the carried load is derived from the offered load by multiplying the latter by one minus the flow rejection ratio. This allows a fair comparison across different SR-TE strategies. Unsurprisingly, the use of SAP leads to the best results in terms of flow rejection ratio, at the expense of slightly longer tunnel lengths. This was expected, as SAP allows more flexibility in the chosen path. The reduction in flow rejection ratio is most notable for low traffic loads (Fig. 4a), whereas the increase in tunnel length is most prominent at higher loads, when the network is more congested (Fig. 4b).

Subsequently, Fig. 5 represents the results for TI-LFA rerouting when facing single link failures, using the same traffic data and the same topology as in the previous case. Results for four different SR-TE approaches are shown, corresponding to the two path selection and two encoding methods discussed in Section II. To distinguish between SP and SAP we again use blue and red colours, with darker colour tones (and circular markers) for node encoding and a lighter colour tones (and triangular markers) when SR-LEA encoding is used. In each plot, the x-axis represents the normalised *carried* traffic load in the network. Fig. 5a shows the *success rate*, i.e., the percentage of times that TI-LFA succeeds in rerouting a flow when facing link failure. As expected, the success rate decreases steadily as the network gets more loaded, irrespective of the SR-TE strategy used. This can be easily explained because in that situation, the backup path is more likely to overload the links of the new path to the destination. The next plots elaborate on the cases where the recovery was successful. Fig. 5b and 5c represent the path stretch after using the backup path, expressed in additional number of hops and percentage increase relative to the primary path, respectively. Both



(a) Flow rejection ratio



(b) Average tunnel length

Fig. 4: Comparison of the network performance for SP and SAP for increasing network loads.

figures show a similar decreasing trend as traffic load increases. This is because in more loaded networks, the long backup paths have a higher risk of crossing at least one congested link, causing the entire recovery to fail. Finally, Fig. 5d examines whether traffic flows pass through the same node more than once when rerouted. More precisely, it represents the average number of node repetitions that a rerouted traffic flow faces. We find that the more there is a deviation from the shortest path (here, SAP), and the more labels are included in the segment list (here, node encoding), the higher the risk of such node repetitions.

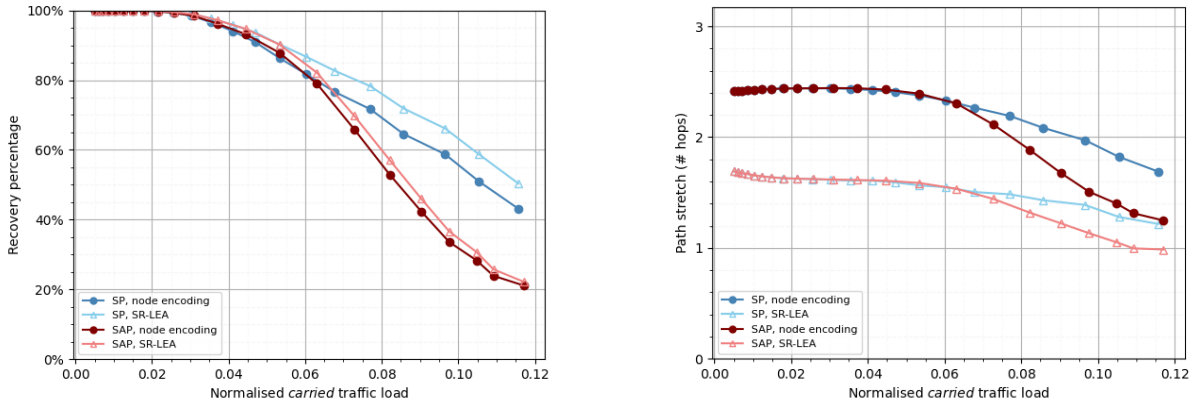
C. Discussion of the Results

As expected, an SR-TE approach like SAP leads to a lower traffic rejection ratio compared to SP routing. Our results also show that this generally does not lead to inferior TI-LFA performance, as, for instance, using SAP even leads to slightly shorter backup paths than for SP. On the other hand, detours where nodes are visited twice are more likely using SAP and SR-LEA, especially at higher network loads. This is because traffic is rerouted to the next segment instead of the destination, and typically a more congested network requires more intermediate segments to specify a SAP. This is evident from the results using the full 'node encoding', where node repetitions happen more often than with SR-LEA. In those cases, the segments to route around the failed link intersect more frequently with the original route.

Therefore, it could be interesting to examine if we can remove unnecessary detours in the rerouted path rapidly and without invoking the (slow) control plane. One suggestion could be to perform some additional post-processing locally in the data plane, at the time of pushing the pre-computed TI-LFA repair list on top of the remaining segment list. The node that activates the repair list could check if there are duplicates in the concatenated repair and segment list. If this is the case, intermediate labels become obsolete and could simply be deleted. The plausibility of this approach will however strongly depend on the labels used to encode the primary *and* backup path. We see the study of the optimal configuration of the TI-LFA repair lists, tailored to the primary paths, as a topic for future research.

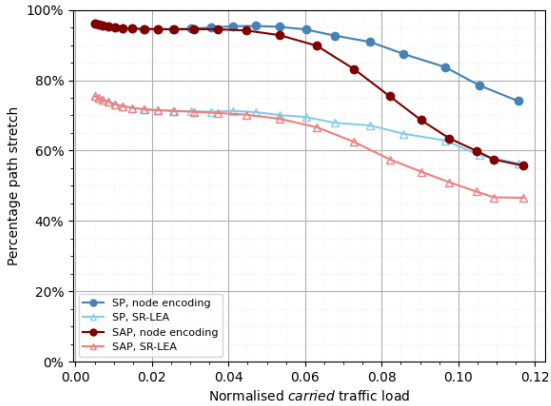
V. Conclusions and Future Work

This paper studied the efficiency of Fast ReRouting in the data plane, before the control plane convergence kicks in. We restricted our study to the specific case of TI-LFA protection against single link failures in segment routing networks for one topology. While Fast ReRoute methods have been in place for a long time and TI-LFA guarantees topological connectivity after a failure, much less attention was paid to the trade-off between their level of resiliency and resource overheads in terms of load and latency. To examine if the 100% failure coverage of TI-LFA comes at the price of inefficient use of the available network resources, we set up a simulator to steer SR-TE traffic through a network and periodically simulate link failures. Our results quantified how the SR-TE methods applied in the network, i.e. the establishment and encoding of SR paths, influence the efficacy with which TI-LFA reroutes traffic to its destination.

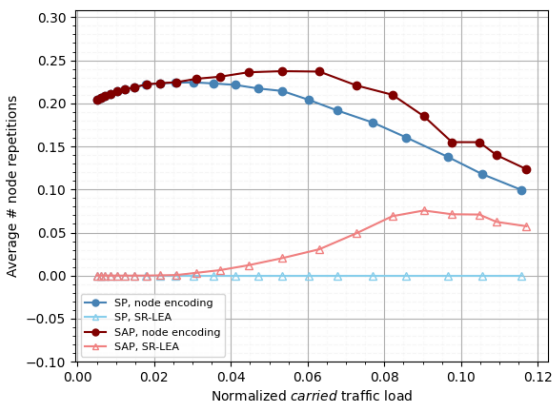


(a) Successful recovery rate

(b) Path stretch



(c) Percentage path stretch



(d) Repeated visits of the same node

Fig. 5: Analysis of the TI-LFA backup paths for different SR-TE strategies and for increasing network loads.

Our experimental simulations also revealed in which conditions the performance of TI-LFA is most harmed. The main inefficiency we observed is the repeated visits to the same node along recovery paths, which leads to increasing congestion and latency in the network. We believe that our observations open interesting directions for future research. As our next steps, we want to analyse our observations across various network topologies and explore how operations in the data plane can help TI-LFA reroute traffic more efficiently for different SR-TE approaches, while balancing generality versus performance.

References

[1] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, Yashar Ganjali, and Christophe Diot, "Characterization of failures in an operational IP backbone network," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 749–762, 2008.

[2] Thomas Holterbach, Edgar Costa Molero, Maria Apostolaki, Alberto Dainotti, Stefano Vissicchio, and Laurent Vanbever, "Blink: Fast connectivity recovery entirely in the data plane," in *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, Boston, MA, Feb. 2019, pp. 161–176.

[3] Alia Atlas and Alex D. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," RFC 5286, Sept. 2008.

[4] Marco Chiesa, Andrzej Kamisiński, Jacek Rak, Gábor Rétvári, and Stefan Schmid, "A survey of fast-recovery mechanisms in packet-switched networks," *IEEE Communications Surveys Tutorials*, vol. 23, no. 2, pp. 1253–1301, 2021.

[5] Clarence Filsfils, Nagendra Kumar Nainar, Carlos Pignataro, Juan Camilo Cardona, and Pierre Francois, "The segment routing architecture," in *2015 IEEE Global Communications Conference (GLOBECOM)*, 2015, pp. 1–6.

[6] Stephane Litkowski, Ahmed Bashandy, Clarence Filsfils, Pierre Francois, Bruno Decraene, and Daniel Voyer, "Topology Independent Fast Reroute using Segment Routing," Internet-Draft draft-ietf-rtgwg-segment-routing-ti-lfa-08, Internet Engineering

Task Force, Jan. 2022, Work in Progress.
[7] Clarence Filsfils, Ketan Talaulikar, Daniel Voyer, Alex Bogdanov, and Paul Mattes, "Segment Routing Policy Architecture," RFC 9256, July 2022.
[8] Pushpasis Sarkar, Stephane Litkowski, Bruno Decraene, Clarence Filsfils, Syed Raza, and Martin Horneffer, "Operational Management of Loop-Free Alternates," RFC 7916, July 2016.
[9] Rabah Guedrez, Olivier Dugeon, Samer Lahoud, and Géraldine Texier, "A new method for encoding MPLS segment routing TE paths," in *2017 8th International Conference on the Network of the Future (NOF)*, 2017, pp. 58–65.
[10] Norman Matloff, "Introduction to discrete-event simulation and the SimPy language," *Davis, CA. Dept of Computer Science*, 2008.
[11] C. Filsfils and K. Michielsen, "Segment routing topology independent LFA," https://www.segment-routing.net/tutorials/2016-09-27-topology-independent-lfa-ti-lfa/, Accessed: January 4, 2023.