Optimal network design of SURFnet8, using TI-LFA and Segment Routing

Peter Prjevara & Fouad Makioui

Research Project 2 - RP2 - System and Network Engineering - SNE

***Abstract*—In this project we set out to provide recommendations to SURFnet on optimum implementation of Topology Independent Loop Free Alternate (TI-LFA) based backup path precomputation on the SURFnet8 network. We have selected a subset of the SURFnet8 topology, and simulated it with 8 Juniper routers. We prove that TI-LFA provides significant performance improvements to service recovery. We observed that an additional hop can negatively effect both IGP and TI-LFA, but further testing with TI-LFA would be required to prove this. We observed large convergence delays if both the primary and TI-LFA backup paths failed simultaneously. We have identified that maximising the Equal Cost Multipaths (ECMPs) in the network topology are beneficial, especially if TI-LFA is used. Carefully combined with fate-sharing, the benefit of maximum number of ECMPs can be further increased. We advise against the use of the node protection feature, if only relying on the information of a failed link. We make recommendations to SURFnet on how to achieve higher amounts of ECMPs within their network. Through a simulation, we have proven that our suggested metrics significantly increase the number of ECMPs within the SURFnet8 topology.**

I. Introduction

Internal Gateway Protocol (IGP) based reactive network recovery is an effective, proven way stabilise networks in case of failures. It is affected by several factors and domain wide convergence of new paths can take a significant amount of time [9] (up to 1000ms [4]). The main factors that affect this time are the time it takes for the failure to be detected (< 50ms [3]), and how fast the new information is propagated across the network devices.

New methods such as *locally computed backup paths*, and faster failure signalling methods such as Bidirectional Forwarding Detection (BFD) [19] with a 1ms sending rate are explored by researchers to improve the speed of recovery. Some of these methods are also used in production such as "fast hellos", or slower rate BFD [5]. The locally computed backup paths added to the Forwarding Information Base (FIB) enable the routers to swiftly switch to the backup paths when a failure is detected. The new primary path after domain-wide convergence of the IGP might be different from the locally computed backup path, and this can result in multiple path transitions [7]. This behaviour is not ideal but better than suffering BGP session loss, which can hinder user experience [17]. This project is set out to find an optimum network design of the newly built SURFnet8 topology by understanding the requirements of SURFnet, examine novel Segment Routing (SR) [6] based locally computed backup path mechanisms such as Topology Independent Loop Free Alternates (TI-LFA) [11], and identify how the goal of achieving lower than 50ms service recovery on failure can best be supported with the help of these new technologies.

A. Reading Guide

In the upcoming section Section II - SURFnet8, we discuss the characteristics of the SURFnet8 topology. Afterwards, in Sections III - Reactive and Proactive Path Recovery and Section IV - Segment Routing, context is provided for understanding TI-LFA, by discussing the differences between reactive and proactive path recovery solutions, and offering an insight into the workings of Segment Routing. In Sections V - The Evolution of the Loop Free Alternate Fast Reroute concepts, VI - Fate-sharing [8] and VII - Link or Node Protection, we summarise the features of TI-LFA, to provide an introduction for our Research Questions, detailed in Section VIII. In Sections IX - Method of Research, XI - Discussion and XII - Conclusion, we detail our results and discoveries and conclude our research questions. Finally in XIII - Future Work, we discuss the work that we believe would be nice to conduct to gain further insights into the working mechanisms of TI-LFA.

II. SURFnet8

SURFnet8 is the new infrastructure that SURFnet is building to support Dutch institutions by offering top network infrastructure. The new proposed network topology contains several redundant paths to ensure uninterrupted operation of services. The topology of the new SURFnet8 network includes multiple 100 Gbps core links, and 10 high capacity core routers (highlighted by white circles with black edge in Figure 1). Due to their extendibility, these routers are better suited to aggregate traffic, and they are preferred next hops, over the daisy chained routers. Hereinafter, the links between these routers are referred to as core links. Several different node types are used in the topology with high performance Juniper MX series routers. As the daisy chained routers are more expensive to expand with additional links, they should only receive traffic that is destined to their regional links, and should not take part in transiting flows between core routers. This

preference also applies to situations where node or link failures occur.

To achieve this goal, SURFnet proposed to have IGP cost 5 for the core links and IGP cost 20 for the daisy chains. Originally, these metrics were proposed to be the same across all core links, and the same applies to the daisy chained links.

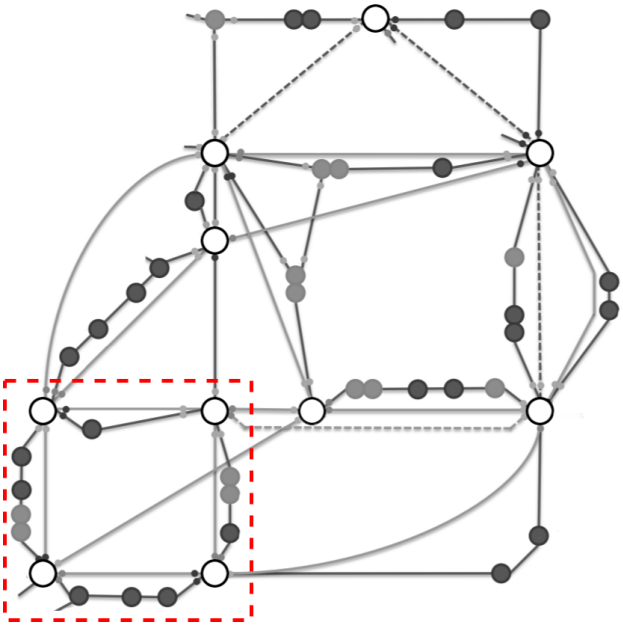

Fig. 1: The SURFnet8 topology, with core routers highlighted by the white blobs. The light grey lines represent the core links on the picture, with the lower proposed IGP metric of 5. The highlighted section is the focus of our research, which we simulated on a testbed depicted in Figure 7, Section IX - Method.

III. Reactive and Proactive Path Recovery

IGP convergence takes a considerable amount of time, as each step within the process introduces a delay in reaching consensus of optimal paths to be used [9]. It is a *reactive* approach - once failure is detected, the nodes in the network start the process of propagating the new link state information across the network. The current IGP protocols are robust, but in some cases aren't fast enough [2]. As an example, IGP convergence can cause BGP session loss when service recovery can potentially take an additional 3 minutes [17], if BGP has to re-establish peering sessions from ground-up. This outage can be unacceptable for real-time services such as Voice over IP (VoIP) or cloud based applications and services. To further reduce service recovery time, and avoid potential BGP session loss, the *proactive* methods called Fast Reroute (FRR) local repair concepts were introduced as early as 2010 in RFC 5714 [5], [13], [2].

Several FRR concepts have been proposed since, and some have been added as an improvement to IGP implementations to enable localised calculation of backup paths during domain-wide flooding of link state information. The existence of pre-computed backup paths allows significantly faster service recovery but they are not always optimal [14], [15]. In general, the ability to create backup paths also require additional protocols such as MPLS, RSVP-TE and LDP to be in place, to allow for explicit declaration of repair paths. The Segment Packet Routing in Networking (SPRING) or Segment Routing mechanism is proposed and implemented by routing vendors to alleviate the need for using additional protocols such as RSVP-TE and LDP, and providing a way of MPLS label distribution relying on the existing IGP mechanisms.

IV. Segment Routing

Segment Routing is a source routing paradigm [6], "that envisions the networks as topological subpaths, also called segments". It enables an ingress node to send a packet along a specific path by adding a set of instructions to the packet. The instructions are imposed in the form of a list of Segment Identifier(s) (SID). SIDs can be separated into three groups. Every node in the domain will be assigned a *Node ID*, and every link in the domain will be assigned an *Adjacency ID*. Adjacency IDs are locally assigned by the nodes, but they are distributed domain-wide with alongside the Node ID. This allows the ingress router to select a specific adjacency instead of a particular node at the time of path selection. The third group of SIDs is *Prefix IDs*, but these are only relevant in the context of BGP [16], and so they are not discussed further in this paper. The Topology Independent Loop Free Alternate is an FRR concept that leverages the Adjacency IDs from the Segment Routing concept. SR and TI-LFA often mentioned together in explanatory articles of TI-LFA [18]. Combining TI-LFA with link fate-sharing techniques, should make it possible to achieve lower service recovery speeds and maintenance costs on the SURFnet8 network.

V. The Evolution of the Loop Free Alternate Fast Reroute concepts

The Loop Free Alternate (LFA) [12] backup path calculation algorithm enables only selection of a next hop as a backup path. In the scenario depicted in Figure 2, R2 has a higher metric path through R3 to R5 then what it has through R1 towards the destination. Hence, when the link between R1 and R6 breaks, a packet sent from R1 to R2 with destination R5 will bounce back to R1 and packet loss, due to looping will occur, until the domain-wide convergence has finished. With some specific topologies, this might also result in sub-optimal backup path selection at the Point of Local Repair (R1 in this case), or worse, a failure of 100% coverage [12].

Remote LFA (rLFA) [1] improves on the situation by introducing the terms *P-space* and *Q-space*, where *P-space* refers to a "set of routers that can be reached from

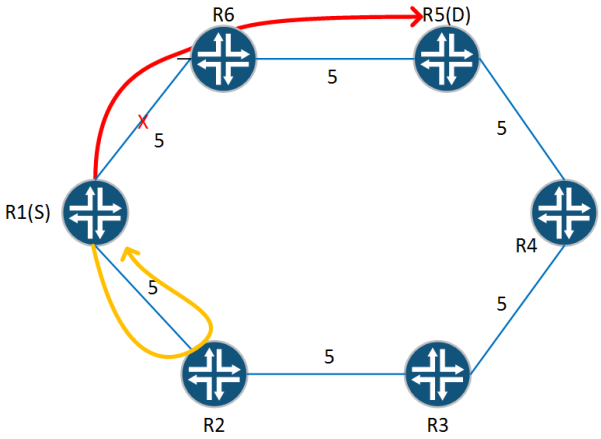

Fig. 2: Ring topology with LFA.

source router without traversing the failed component" and *Q-space* is a "set of routers from which destination can be reached without traversing the failed component" [12]. Allocation of nodes to P and Q spaces are based on link cost. The extended P space first depicted on Figure 3 simply refers to the union of all nodes that are reachable from R1, without traversing the failed link. Remote LFA allows selection of a backup path through a node residing in both P and Q spaces (in the so-called *PQ-space*). It is capable of doing so by creating an MPLS LSP tunnel to a node in the PQ space (dashed line in Figure 3), and pushes the MPLS label of the node the packet before sending. Now, since the packet has reached a node that has a lower cost path to R5, it will find its path using standard IGP techniques.

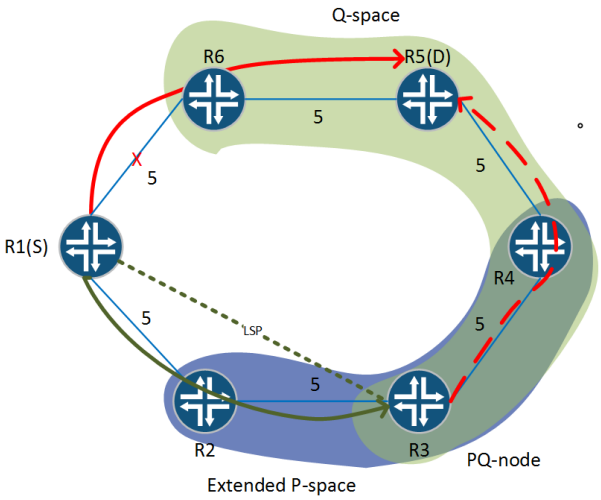

Fig. 3: Ring topology with remote LFA.

Topology Independent LFA (TI-LFA) extends this concept further, by allowing for backup path selection even if *PQ-space* does not exist in the topology, due to too high metrics. Using Adjacency SIDs, TI-LFA is able to install an explicit repair path and reach a node within the *Q-space*, even if there is no overlap between *P-space* and *Q-space*. This happens when metrics are not homogeneous in a domain, and there exist a significantly high metric on a link between two routers. See Figure 4 for an example. TI-LFA is also designed to select the optimum backup path, which must be the same as the IGP path after convergence (referred to as post-convergence path). As TI-LFA is now implemented it allows SURFnet to consider it for its configuration on the new SURFnet8 infrastructure [7].

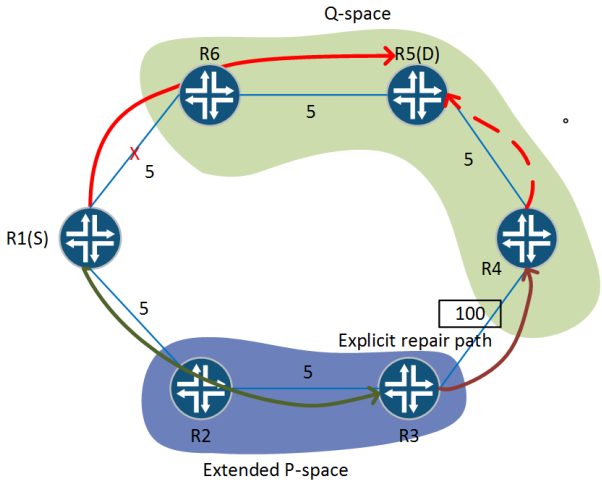

Fig. 4: Example topology without PQ space, due to metric between R3 and R4 increased to 100.

Besides the important feature of being able to use the Adjacency IDs of SR to bridge the gap between P spaces and Q spaces, TI-LFA is also able to take advantage of fate-sharing and node protection. These features will be briefly discussed in the next sections. We also assessed usefulness of these for the SURFnet8 topology through experiments, and we discuss the results later in Section X - Experiments.

VI. Fate-sharing

Coarse / Dense Wavelength Division Multiplexing (CWDM / DWDM), makes it possible for several separate interfaces to share the same physical fibre. There is a possibility that either the common CWDM cable breaks or a line card that contains several interfaces fails. In any of these cases all interfaces share the same fate - they become unavailable simultaneously.

In this case it can be beneficial to disallow the routers to select a fate shared interface as FRR backup path. In Figure 5, the benefit of fate-sharing is depicted. Even though the path through R2 has a lower cost, R1 is selected as primary next hop to ensure the CWDM fate-sharing links are not used as backup paths. Fate-sharing uses the concept of groups [14], that are created locally on each router, containing a cost assigned to the group. This cost value is added to the standard IGP cost of each link, to make these links less favourable during possible

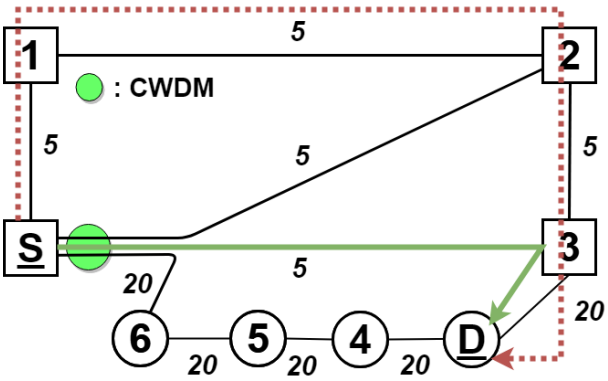

Fig. 5: Fate-sharing example

backup path selection. The links are still considered, but with an increased cost. This information is not shared with other nodes, so the are node-local only.

VII. Link or Node Protection

The effectiveness of the locally computed backup paths also depend on whether a whole node, or just a link failed. If a whole node within a network topology fails, it is possible that several paths become unavailable simultaneously, as they might all traverse the particular node. In case of the example depicted in Figure 6, in case R3 fails, the more costly backup path should be chosen, as that will be the least costly path after IGP convergence.

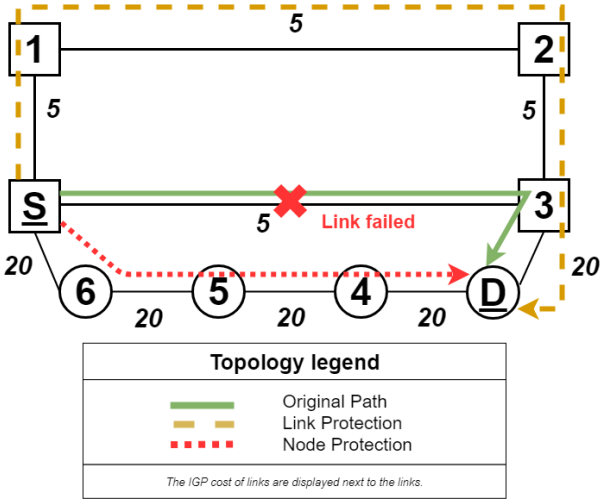

Fig. 6: Example of Node / Link protection

A. Strict or loose node protection

Juniper implements two ways to achieve node protection. Strict protection means that all the links that are connected to a node are excluded from the proactive backup path calculations entirely.

Loose node protection works similarly to fate-sharing: a cost can be assigned to the node, which cost will be added to the standard cost of each link connected to the node. This way those links will be less favourable at the time of backup path selection.

VIII. Research Questions

Our project is set to discover the optimum configuration of the implementation of TI-LFA based path precomputation mechanisms to best suit the SURFnet8 infrastructure. The features described in Section VI - Fate-sharing and Section VII - Link or Node Protection are implemented and their effectiveness is tested, and an optimal configuration will be proposed. As the path selection mechanism of TI-LFA depends on IGP metrics, the currently proposed IGP metrics of the SURFnet8 infrastructure will be thoroughly assessed and possibly adapted to support fast backup path selection. In order to find the optimum configuration for SURFnet, we have outlined the following research questions to be answered through experiments:

- How do different configurations of TI-LFA compare if node failure, link failure, or both happen at the same time, on multiple nodes?
  - How does the SURFnet8 specific IGP metric costs effect the above outcomes?
- Is fate-sharing necessary for all links that share the same optical path, or would TI-LFA be sufficient to provide efficient backup path coverage?

IX. Method of Research

To conduct our experiments we have built a testbed of 8 Juniper routers. To be able to introduce realistic link failures, we remove the fibre cable from different interfaces. We have conducted each experiment 20 times unless stated otherwise. After every 10 experiments we cleaned the fibre connection and the optic that resides in the router too.

A. Limitations

To effectively simulate interface failures, we manually remove cables from the router interfaces. This introduces some uncertainties and we can not guarantee 100% accuracy at failure induction. This should be taken into account when one is reviewing results.

No other topologies with different Source and Destination routers were tested, due to the limitations imposed by the project schedule.

B. Testbed

A physical test bed of SURFnet is used to simulate a part of the full SURFnet8 topology. The dashed lines in Figure 1 depicted the part of the topology we have selected to simulate. The similarities between our simplified topology depicted in Figure 7 and the focus point highlighted with dashed lines in Figure 1 are the core routers and the core links with lower IGP metric of 5, a 4 router daisy chain with higher IGP metric

of 20 and the existence of a CWDM, optical cable sharing connection, starting from the Source router (S) connected to R2, R3 and R6. The crosslink between S and R2 allows creation of multiple backup paths towards the Destination router (D). The test topology also allows for introducing a Single Point of Failure (SPOF) through removing the common cable from the CWDM cable tray, and this way many different types of failures can be tested. A detailed list of nodes and software versions used in the testbed can be found in Appendix 1.

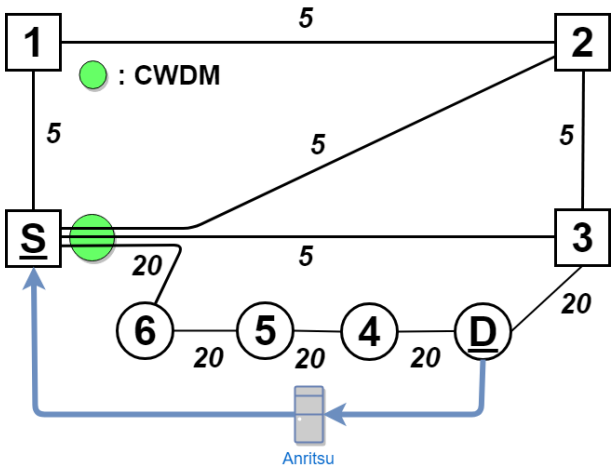

Fig. 7: Simplified diagram of the test topology

Both the rectangular positioning of the core routers and the routers in the daisy chain are appearing several times within the SURFnet8 topology as can be seen in Figure 1. This means that the results conducted on our test environment can be extrapolated to the whole network.

C. Recovery measurements

An Anritsu MD1230B Data Quality Analyser (hereinafter the Anritsu) is used to measure packet loss during path restoration period. Between the input and output interfaces of the Antritsu, there is a Layer 2 EVPN tunnel service set up.

We set the Anritsu to send 10,000 frames per second (f/s). This way we achieve millisecond accuracy measurement of restoration, with maximum error of approximately 0.05%. We decided at this rate after testing both with 1,000 f/s and 1,000,000 f/s rates. At these rates we observed 0.5% and 50% error respectively (see Appendix 2 for details). The errors are introduced as the Anritsu is not capable of upholding the given f/s speed and it occasionally reduces to 9,995 f/s.

The 0.05% error affects the accuracy of our measurements per experiment. Based on an average experiment time of 10 seconds, this means that the results will have to be interpreted with up to 5ms error.

D. Routing table analysis

Simultaneously to measuring packet loss, we query the routing table of the Point of Local Repair (PLR) router, every 20ms for 1 minute. We achieve this by using MobaXTerm terminal application. This results in 3,000 queries / minute, which from we selected the unique items using a simple Python script, that filters all unique routes and presents them in a terminal output format.

E. Topology simulation

As explained in Section V - The Evolution of the Loop Free Alternate Fast Reroute Concepts, TI-LFA path calculation relies on the IGP metrics. We expect the experiments provide us with insight on whether the proposed metrics are indeed effective. The SURFnet topology and the proposed metrics are simulated using another Python script and networx library. The *networkx* library allows simulation of networks with nodes and links between them, with costs assigned. The Python script relies on this library to be able to calculate the shortest paths between every node in a network, see Appendix 5 for the source code.

X. Experiments

Table I outlines the experiments we performed. The subsections of this section describe each experiment in detail, the expected outcome of measurements and routing table analysis and the observed outcome of those. The list of actual outcomes can be found in Appendix 1.

Name	Variation
Baseline SR	IGP only
	TI-LFA enabled
Baseline SR with extra hop	IGP only
	TI-LFA enabled
Multiple link failures	With a single backup path
	With equal cost multi paths
	With fate-sharing enabled
Link / node protection	
Metric optimisation	

TABLE I: Conducted experiments

A. Baseline SR - IGP only

First we wanted to measure the baseline IGP recovery on our topology. As illustrated in Figure 8, we removed the connection between S and R3, from the CWDM cable tray (leaving 2 out of 3 links of the CWDM untouched). Based on our literature review explained in Section I - Introduction, we expected the recovery to happen under 1000ms, and the results concurred: **the average speed of IGP convergence was measured to be 304ms**.

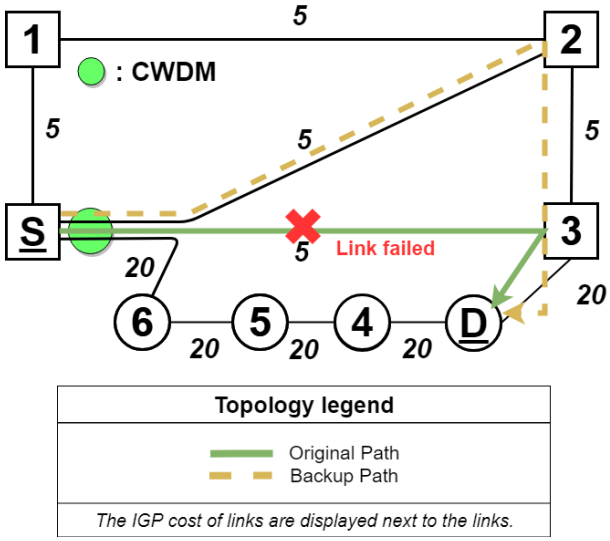

Fig. 8: Baseline SR experiment. Recovery path for IGP and TI-LFA is the same, highlighted with the dashed line.

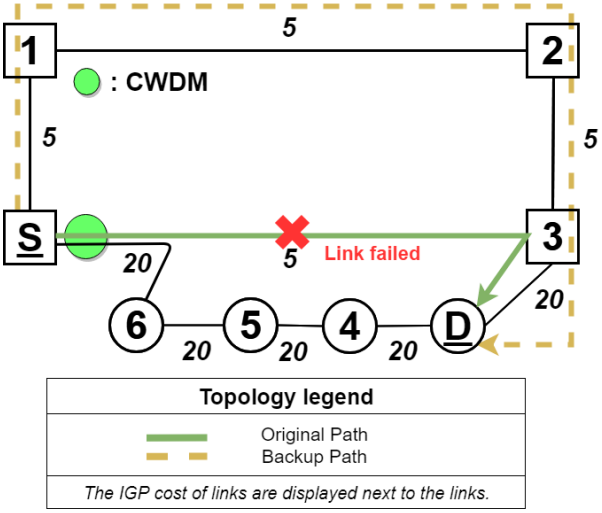

Fig. 9: Baseline SR experiment with extra hop. Recovery path for IGP and TI-LFA is the same, highlighted with the dashed line.

B. Baseline SR - TI-LFA enabled

We then enabled TI-LFA based proactive backup path calculation on the Source router. We observed how the TI-LFA precomputed backup path that is illustrated in Figure 8 is added to the routing table as backup path. We expected the recovery to happen within 50ms, and the results concurred: **the average speed of TI-LFA based recovery was measured to be 26ms**.

C. Baseline SR with extra hop - IGP only

To see how different topologies effect convergence times, we have disabled the crosslink on our topology (see Figure 9, for description). This resulted in the backup path to traverse R1 first, before reaching R2, adding an extra hop to the backup path. We disabled TI-LFA and repeated the same experiment, as in the previous sections. We measured the effect of the extra hop in the path and compared it to the baseline IGP convergence times. We observed a **55ms increase** in comparison to what was observed in Section X-A - Baseline SR - IGP only. Based on the information gathered through the literature review, this result was expected. The reason is that due to the removal of the shorter link, the travel time of the link state packets increased slightly, in addition to the calculation and path addition times on the routers due to the introduced extra hop.

D. Baseline SR with extra hop - TI-LFA enabled

In the next experiment we enabled TI-LFA again. We observed how the TI-LFA precomputed path that is depicted in Figure 9 is added to the routing table as backup path. We expected the recovery to be around the same as with the previous experiment with TI-LFA, as this technology precomputes the backup path. **The average recovery time of 20 experiments increased by 5ms**. We observed an additional label popped to the packet in the routing table. We hypothesise that this is the reason for the increase, that an additional label that has been popped by TI-LFA, but further testing would be required to verify this idea.

E. Summary of baseline experiments

The bar chart in Figure 10 depicts the results explained in the previous sections.

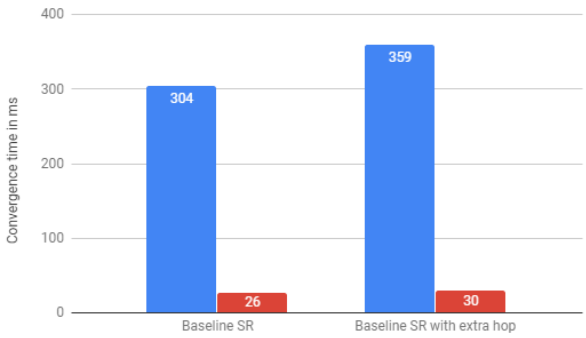

Fig. 10: Comparing IGP to TI-LFA recovery times with and without an extra hop in the topology

TI-LFA always outperforms the IGP protocol as expected, and both protocols perform within their expected limits [4], [3]. It is unexpected, that an additional hop will also affect the precomputed path convergence time even so slightly, even on a short distance as in the testbed. It seems that the **convergence time of TI-LFA is affected by an additional hop**, but further experiments with additional SIDs imposed would be needed to verify this observation.

F. Multiple link failures - With a single backup path

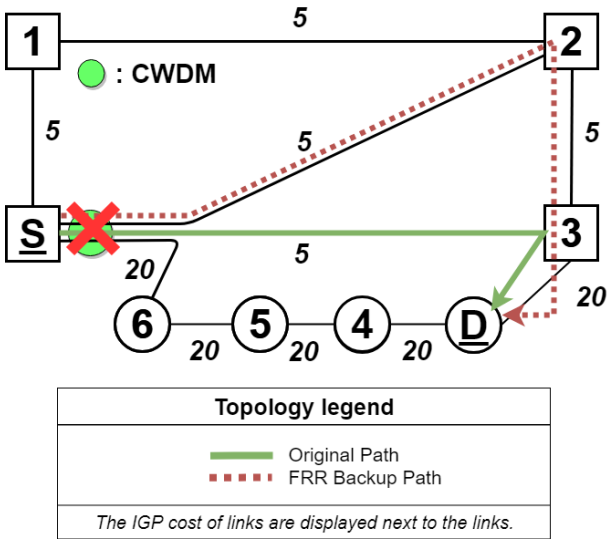

Fig. 11: In this experiment, both the primary path and the precomputed backup path fails simultaneously.

Next, we wanted to see how what the convergence time is affected if both the primary and TI-LFA pre-computed backup path fails simultaneously. We induced this situation by allowing the backup path to share the same CWDM link with the primary path, and remove the main CWDM link that carries all signals from the CWDM duct. This situation is depicted in Figure 11.

The expectation was that we will see convergence times similar to the IGP rates displayed in Section X-E - Summary of baseline experiments. **The results showed a large increase in average of convergence time, with the average of 702ms out of 13 experiments.** The largest sample was 2879ms, which is more than 10x of what we have recorded for IGP convergence. The calculated standard deviation was 692ms. We believe the large difference between the results was due to the uncontrollable variations in failure detection, caused by our inaccurate fault induction described in Section IX-A - Limitations.

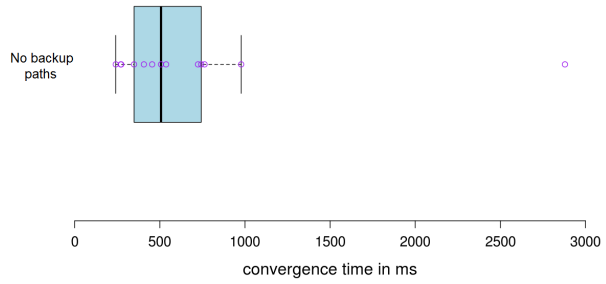

Fig. 12: The results of multiple link failures with a single backup path

G. Multiple link failures - With Equal Cost Multi Paths

JunOS allows for configuring up to 8 backup paths with TI-LFA. In the book MPLS in the SDN era book of Monge et al, the selection criteria that is used by the TI-LFA algorithm is documented [9]. The document stated that first and foremost, Equal Cost Multi Paths (ECMPs) are preferred as backup paths. We have increased the cost of the crosslink to 10, as depicted in Figure 13, and we have observed that an additional path through R1 was added to the routing table as backup path.

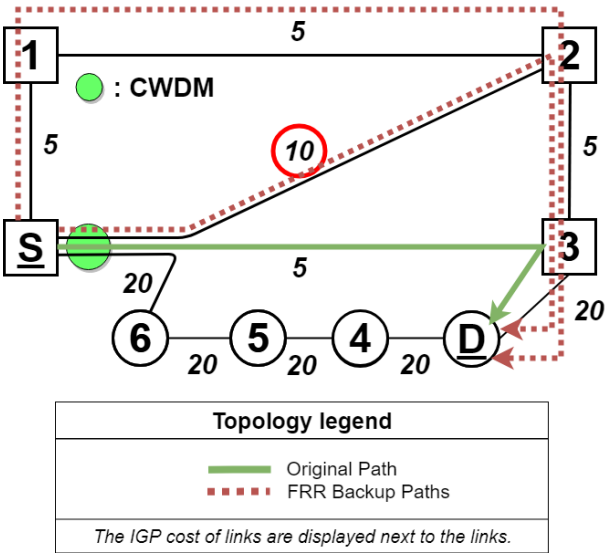

Fig. 13: When the cross link cost was increased to 10, multiple backup paths were added to the routing table.

We have repeated the experiment described in the previous section, and expected large improvements with the recovery times. The results of the experiments concurred with our expectations, when we observed **an average recovery time of 108ms** and a reduction of the standard deviation between experiments to 172ms. This reflects a more stable recovery mechanism, compared to the one in the previous experiment. The 108ms recovery time is still quite large if compared to the observations in Sections X-A, X-B *Baseline SR with / without TI-LFA*, but the significant improvement suggests that **having more ECMPs in a network using TI-LFA is beneficial.**

H. Multiple link failures - With fate-sharing

For our final experiment with the Anritsu, we have decreased the crosslink metric to the original 5, and created a fate-sharing group that included the CWDM links on the Source Router (S). First we did not configure a cost for the group, which meant that the default cost 1 was used by the algorithm. For the purpose of the backup path calculations this meant that crosslink was increased to 6, hence the crosslink path through R2 has still has a lower cost (with total path cost of

31), compared to the path through R1 (total path cost 35). This way, the routing table is not different from the situation that is depicted in Figure 11.

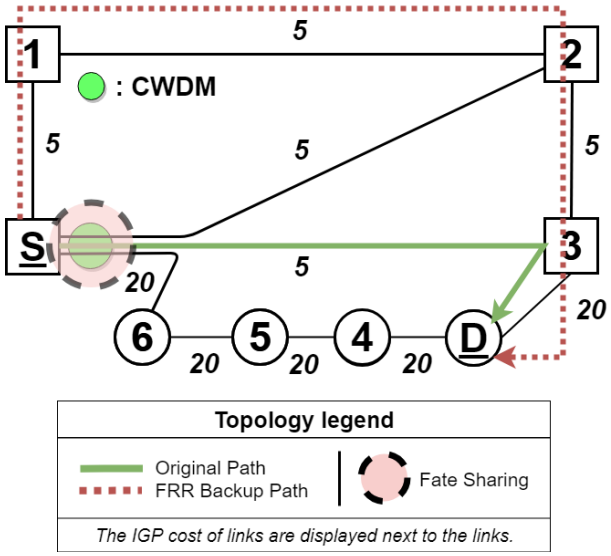

Fig. 14: When fate-sharing is enabled, the fate-sharing links' metrics are increased by a configurable amount (cost). This cost is added to the link when the links are considered at the time of the FRR backup path precalculation.

We then increased the fate-sharing group cost to the maximum 65535 [8] (see Appendix 3 for the configuration), and observed the update of the backup path within the routing table. We then conducted the same experiment as described in Section X-F - Multiple link failures - With a single backup path (removing the common CWDM line connection, causing all links to fail at the same time). Our expectation was that the recovery speed will be similar to the ones we have seen in Section X-D - Baseline SR with extra hop - With TI-LFA enabled. It was visible after a couple of experiments that our assumptions were correct, and the precomputed backup path allows for under 50ms convergence. As the backup path is exactly the same as what we have created in the aforementioned Section, we did not repeat the experiments more times, **and we are assuming that the recovery times of Section X-D - Baseline SR with extra hop - With TI-LFA enabled apply.**

I. Summary of multiple link failure experiments

When paths in the network that are used as backup paths for each other fail simultaneously, the network can be affected for longer than standard IGP convergence times. We believe that the large deviation between measured results are due to the unpredictable nature of our fault induction explained in Section IX-A - Limitations. Further testing would be required to understand the root cause of the longer than IGP convergence times.

It is clearly visible that if multiple backup paths are available the recovery times significantly improve and become more predictable. This point proves the importance of having the maximum number of ECMPs, if using TI-LFA on a network.

If it can be predicted that interfaces are going to fail simultaneously, it is even more beneficial for the recovery times if fate-sharing is implemented besides multiple backup paths.

The summary of the last two experiments are depicted on the chart in Figure 15.

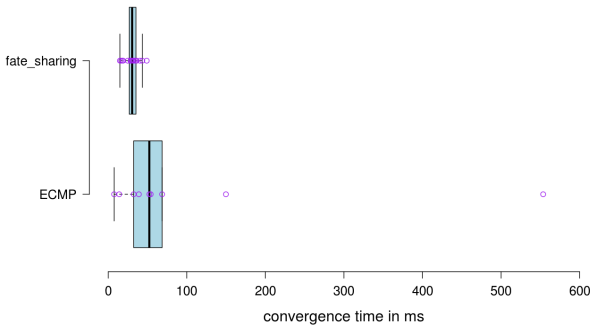

Fig. 15: The results of the experiments from the previous two sections. TI-LFA with fate-sharing is very effectively protecting against multiple link failures, compared to only having multiple backup paths, or no backup paths at all.

J. Link / node protection

Link protection is the default behaviour of TI-LFA. In order to fully answer our first research question, we configured node protection on the Source Router (see Appendix 4 for the configuration). As a preparation, we recreated the multiple backup paths known from Section X-G *Multiple link failures - With Equal Cost Multi Paths*, by increasing the crosslink metric to 10. Then we enabled *strict* node protection, and observed how the multiple backup paths were removed from the routing table, reducing the available backup paths to one. This situation is depicted in Figure 16.

It is a harsh decision to consider the whole router failed, based on no other information but that a single link is down. Due to the fact that node protection can negatively influence backup paths, **the need for such harsh exclusion of links should be carefully considered**, and only dynamically implemented if additional information dictates (for instance in case known power outages effecting a certain area of the country).

K. Metric optimisation

Metrics affect the cost of paths, which has important implications at the time of backup path calculations, as we have seen in Section X-G - Multiple link failures - With Equal Cost Multi Paths. In order to achieve

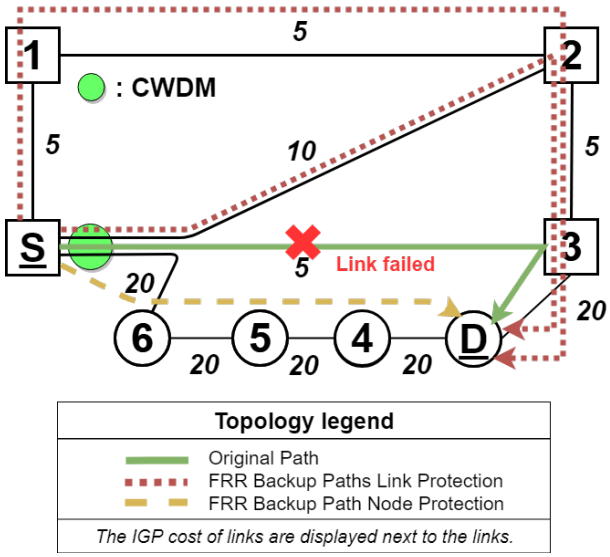

Fig. 16: In our topology, enabling node protection disabled two backup paths, and replaced it with one.

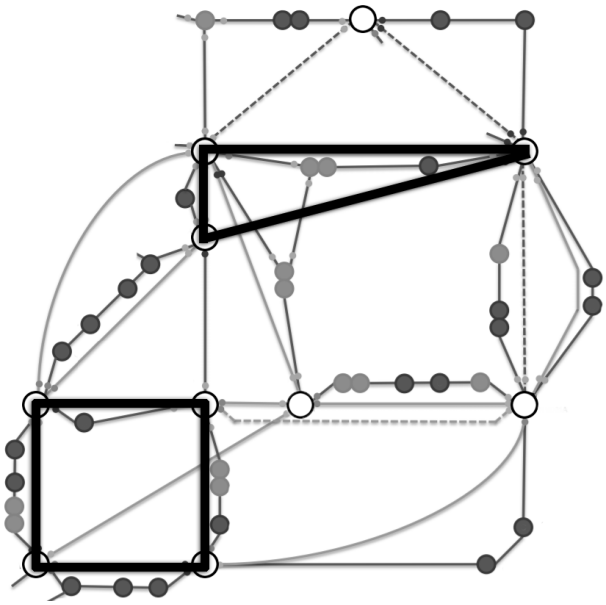

Fig. 17: Example of triangles and rectangles of links in the SURFnet8 topology.

the highest number of ECMPs within the SURFnet8 topology, we have followed this simple argument: Most daisy chained links have only a single connection going through them, so the backup paths for nodes on daisy chained links are dependent on the core routers. As a result, the goal reduces to finding the highest number of ECMPs within between the core links and core routers. We took a close look at the topology, and identified rectangles and triangles of links between the core routers. It can be easily seen that if equal metrics are used on the core links, the rectangles are already providing ECMPs between 4 nodes. The same can be achieved with triangles if the cost on the longest edge of the triangle is duplicated (as we did in the case when we increased the crosslink value in Section X-G - Multiple link failures - With Equal Cost Multi Paths). To prove that this statement is correct we recreated a virtual topology of the core routers and core links with the Python script described in Section IX-E - Topology simulation. We ran the simulation applying equal metrics to all the links that resulted in 268 equal cost paths (primary and backup). **By identifying triangles and increasing the duplicating the metric on the longest edge this number increased to 423**.

XI. Discussion

SR simplifies the distribution of labels and allows for dynamic allocation of Adjacency IDs. TI-LFA leverages this technology to precompute backup paths, that can traverse any link on the network, regardless its cost using an Adjacency ID. The SURFnet8 topology in a normal state has several backup paths and symmetric metrics, that is providing a homogeneous environment. Hence, the feature of TI-LFA, that is able to bridge a possible *P Q space gap* does not benefit the SURFnet8 topology at the normal state. The topology is made up of subsets of daisy chained nodes and core nodes - our testbed simulated one such topology and found that no more than 2 labels will be required at any time, and Adjacency SIDs are also not required. This finding is also supported by the research done by Cisco and Orange [10]. Regardless, TI-LFA is beneficial to SURF as it reduces configuration time due to its reliance on SR, instead of RSVP-TE or LDP.

Even though TI-LFA precomputes the backup paths, the length of the path still seem to affect recovery times. Based on our experiments we hypothesise that longer paths take more time to install as recovery paths, due to the additional labels required to be added to the frames. This finding is negligible, as TI-LFA significantly improves recovery times in general, with measured average recovery times in the ranges of 10s of milliseconds.

Fate-sharing allows for excluding certain links from being considered as backup paths for a certain destination, which is can be beneficial for interfaces that share the same fate. If enabled, fate-sharing might cause the use of sub optimal backup paths, but still keeps the recovery within the ranges of 10s of milliseconds.

It was observed several times during testing that after a failure, that occasionally a 10ms path transition time can happen, when the primary path is reinstalled in the FIB, after repairing the phyisical connection.

We discovered that multiple ECMPs are benefitting the network significantly. Enabling node protection will reduce the number of ECMPs, so the use of this feature has to be carefully assessed. Potential use case of node protection can be if power outages are expected in a certain area, or planned maintenance of a certain node is upcoming.

We provide some insights into how the number of

ECMPs in the topology can be increased, and we prove that our approach works by introducing an additional 155 ECMPs to the topology. Further optimisation techniques should be examined, as the downside of our approach is that it is very local to a certain subset of the topology.

XII. Conclusion

Through this project we have answered the three research questions set out in Section VIII - Research Questions, by analysing the features of TI-LFA through experiments. Our experiments proved that TI-LFA is an efficient way of backup path precalculation, especially if maximum amount of ECMPs exist in a network. We recommend improvements to the proposed metrics by SURFnet and we prove these through a simulation of the SURFnet8 topology. With regards to fate-sharing we conclude that it has clear benefits if applied on lines that fail simultaneously, but we would recommend further testing of corner cases with several PLRs to better understand the usefulness of this feature for SURFnet.

XIII. Future Work

A. Additional test scenarios

As explained in Section IX-A - Limitations, we have only conducted experiments with the specified Source and Destination routers, with the Source router being the Point of Local Repair (PLR), where the backup path installation happens. It would be particularly interesting to see how TI-LFA with fate-sharing reacts on multiple link failures, when more than one failures are introduced in the topology resulting in several PLRs in the network.

B. Label cost estimation

We would like to verify if imposing an additional label on the frame really has an increasing effect on the recovery times. For this a special topology setting would be required with larger metrics, to force TI-LFA to install Adjacency IDs.

C. Bidirectional Forwarding Detection

TI-LFA significantly improves on recovery times, by alleviating the need for waiting for new link state information before achieving path recovery. The switching from primary to backup path happens in milliseconds. The next factor that significantly affects backup path convergence is the speed of error detection. One technology that can improve failure detection is Bidirectional Forwarding Detection or BFD. It is already implemented with BGP and RSVP-TE based LSPs, and it would be interesting to see how the recovery rates of Topology Independent Fast Reroute based on RSVP-TE combined with BFD is comparing to TI-LFA. Based on the literature review [19], we expect a significant improvement within recovery rates.

XIV. Acknowledgements

We would like to thank SURFnet for providing us with state of the art equipment and trust. Special thanks going to Marijke Kaat, Wouter Huisman and Pieter de Boer for their time. Their passion and knowledge about networking in general inspired and helped us a lot. Without their continuous support and immediate availability, this project would have achieved a lot less.

REFERENCES

[1] S. Bryant, C. Filsfils, S. Previdi, M. Shand, and N. So. Remote loop-free alternate (lfa) fast reroute (frr). RFC 7490, RFC Editor, April 2015.

[2] Gábor Enyedi. Novel algorithms for ip fast reroute. 2011.

[3] L Florit. Putting 50-ms in perspective. *Generic Metro Ethernet Network IP NGN Architecture thought Leadership Journal*, 2008.

[4] Pierre Francois, Clarence Filsfils, John Evans, and Olivier Bonaventure. Achieving sub-second igp convergence in large ip networks. *ACM SIGCOMM Computer Communication Review*, 35(3):35–44, 2005.

[5] IETF, 2015. IP Fast Reroute Framework - RFC4090.

[6] IETF, 2017. IETF SPRING Segment Routing DRAFT.

[7] IETF, 2018. Topology Independent Fast Reroute using Segment Routing.

[8] Juniper, 2018. Juniper Fate-Sharing.

[9] Antonio Sanchez Monge and Krzysztof Grzegorz Szarkowicz. *MPLS in the SDN Era: Interoperable Scenarios to Make Networks Scale to New Services*. " O'Reilly Media, Inc.", 2015.

[10] Cisco Networks, 2016. Cisco TI-LFA with Segment Routing.

[11] Cisco Networks, 2017. Cisco TI-LFA.

[12] Juniper Networks, 2015. Juniper TI-LFA.

[13] Juniper Networks, 2017. Juniper Tech Library - Fast Reroute Overview.

[14] Juniper Networks, 2017. JunosÂő OS IS-IS Feature Guide.

[15] Juniper Networks, 2017. Junos Configuration Guide - Understanding Loop-Free Alternate Routes for IS-IS.

[16] Stefano Previdi, Clarence Filsfils, Acee Lindem, Arjun Sreekantiah, and Hannes Gredler. Segment routing prefix sid extensions for bgp. Internet-Draft draft-ietf-idr-bgp-prefix-sid-26, IETF Secretariat, June 2018. http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp-prefix-sid-26.txt.

[17] Kotikalapudi Sriram, Doug Montgomery, Oliver Borchert, Okhee Kim, and D Richard Kuhn. Study of bgp peering session attacks and their impacts on routing performance. *IEEE Journal on Selected Areas in Communications*, 24(10):1901–1915, 2006.

[18] Cisco Systems, 2017. Cisco TI-LFA / Segment Routing.

[19] Niels LM Van Adrichem, Benjamin J Van Asten, and Fernando A Kuipers. Fast recovery in software-defined networks. In *Software Defined Networks (EWSDN), 2014 Third European Workshop on*, pages 61–66. IEEE, 2014.

APPENDIX 1

Router Number	Type	Os version
Source Router	Juniper MX240	18.1R2.5
R1	Juniper MX480	18.1R2.5
R2	Juniper MX204	18.1R2.5
R3	Juniper MX2008	18.1R2.5
R4	Juniper MX204	18.1R2.5
R5	Juniper MX204	18.1R2.5
R6	Juniper MX204	18.1R2.5
Destination Router	Juniper MX204	18.1R2.5

TABLE II: Router overview.

#	Convergence Time	Average	Min	Max
1	310.4	303.795	287.9	396.5
2	327.5	303.795	287.9	396.5
3	295.3	303.795	287.9	396.5
4	298.5	303.795	287.9	396.5
5	297.4	303.795	287.9	396.5
6	287.9	303.795	287.9	396.5
7	288.4	303.795	287.9	396.5
8	304.6	303.795	287.9	396.5
9	300.5	303.795	287.9	396.5
10	290.8	303.795	287.9	396.5
11	295.1	303.795	287.9	396.5
12	312.4	303.795	287.9	396.5
13	289.7	303.795	287.9	396.5
14	297.9	303.795	287.9	396.5
15	396.5	303.795	287.9	396.5
16	292.8	303.795	287.9	396.5
17	312.1	303.795	287.9	396.5
18	298	303.795	287.9	396.5
19	291.6	303.795	287.9	396.5
20	288.5	303.795	287.9	396.5

TABLE III: Results: Baseline IGP With Crosslink

#	Convergence Time	Average	Min	Max
1	39	107.92	7.5	553.6
2	68.6	107.92	7.5	553.6
3	54.1	107.92	7.5	553.6
4	149.8	107.92	7.5	553.6
5	7.5	107.92	7.5	553.6
6	553.6	107.92	7.5	553.6
7	52.3	107.92	7.5	553.6
8	14	107.92	7.5	553.6
9	32.4	107.92	7.5	553.6

TABLE IV: Results: ECMP

#	Convergence Time	Average	Min	Max
1	337.5	359.49	337.5	407
2	349.7	359.49	337.5	407
3	343.7	359.49	337.5	407
4	394.8	359.49	337.5	407
5	356.3	359.49	337.5	407
6	364.7	359.49	337.5	407
7	338.3	359.49	337.5	407
8	348.8	359.49	337.5	407
9	366	359.49	337.5	407
10	407	359.49	337.5	407
11	398.3	359.49	337.5	407
12	343.9	359.49	337.5	407
13	341.8	359.49	337.5	407
14	345.7	359.49	337.5	407
15	344.1	359.49	337.5	407
16	347.2	359.49	337.5	407
17	355.4	359.49	337.5	407
18	358	359.49	337.5	407
19	373.6	359.49	337.5	407
20	375	359.49	337.5	407

TABLE V: Results: Baseline IGP Without Crosslink

#	Convergence Time	Average	Min	Max
1	17.1	25.855	16.9	37.4
2	26.2	25.855	16.9	37.4
3	27.8	25.855	16.9	37.4
4	20.6	25.855	16.9	37.4
5	34	25.855	16.9	37.4
6	24.5	25.855	16.9	37.4
7	31	25.855	16.9	37.4
8	19.1	25.855	16.9	37.4
9	18.8	25.855	16.9	37.4
10	29.6	25.855	16.9	37.4
11	27.6	25.855	16.9	37.4
12	29.6	25.855	16.9	37.4
13	20.6	25.855	16.9	37.4
14	20.7	25.855	16.9	37.4
15	21.6	25.855	16.9	37.4
16	30.8	25.855	16.9	37.4
17	16.9	25.855	16.9	37.4
18	32.7	25.855	16.9	37.4
19	30.5	25.855	16.9	37.4
20	37.4	25.855	16.9	37.4

TABLE VI: Results: Baseline TI-LFA With Crosslink

#	Convergence Time	Average	Min	Max
1	2879.5	701.95	241	2879.5
2	241	701.95	241	2879.5
3	454.8	701.95	241	2879.5
4	270.5	701.95	241	2879.5
5	762.7	701.95	241	2879.5
6	977.7	701.95	241	2879.5
7	724.2	701.95	241	2879.5
8	348.5	701.95	241	2879.5
9	273	701.95	241	2879.5
10	743	701.95	241	2879.5
11	507	701.95	241	2879.5
12	406.5	701.95	241	2879.5
13	536.9	701.95	241	2879.5

TABLE VII: Results: Multiple Lines Broken

#	Convergence Time	Average	Min	Max
1	14.9	30.215	14.9	49
2	32.8	30.215	14.9	49
3	18.2	30.215	14.9	49
4	32.6	30.215	14.9	49
5	49	30.215	14.9	49
6	28.6	30.215	14.9	49
7	40.2	30.215	14.9	49
8	35.6	30.215	14.9	49
9	29.6	30.215	14.9	49
10	28.5	30.215	14.9	49
11	43.4	30.215	14.9	49
12	16.4	30.215	14.9	49
13	36	30.215	14.9	49
14	28.5	30.215	14.9	49
15	31.3	30.215	14.9	49
16	28.5	30.215	14.9	49
17	24.9	30.215	14.9	49
18	31.3	30.215	14.9	49
19	35.1	30.215	14.9	49
20	18.9	30.215	14.9	49

TABLE VIII: Results: Baseline TI-LFA Without Crosslink

APPENDIX 2

	A	B	C	D	E	F	G
1	Name	Unit1:2:2 Current	Unit1:2:2 Accumulated	Unit1:4:7 Current	Unit1:4:7 Accumulated	Total Current	Total Accumulated
2	Transmitted Bit Rate (bit/s)	11,988,000bit/s	7,071,158bit/s	511,488bit/s	301,891bit/s	-	-
3	Transmitted Bit Rate (%)	0.12%	0.07%	0.05%	0.03%	-	-
4	Transmitted Byte	1,498,500	16,794,000	63,936	716,992	1,562,436	17,510,992
5	Transmitted Frame	999	11,196	999	11,203	1,998	22,399
6	**Error**	**99,90%**		**99,90%**			
7	Transmitted Test Frame	999	11,196	0	0	999	11,196
8	Received Bit Rate (bit/s)	512,000bit/s	301,891bit/s	11,988,000bit/s	7,071,217bit/s	-	-
9	Received Bit Rate (Mbit/s) 1	0Mbit/s	0Mbit/s	-	-	-	-
10	Received Bit Rate (Mbit/s) 2	0Mbit/s	0Mbit/s	-	-	-	-
11	Received Bit Rate (Mbit/s) Other	0.512Mbit/s	0.302Mbit/s	-	-	-	-
12	Received Bit Rate (%)	0.01%	0.00%	1.20%	0.71%	-	-

Fig. 18: Anritsu lost 0.1% of frames at a sending rate of 1,000 f/s

	A	B	C	D	E	F	G
1	Name	Unit1:2:2 Current	Unit1:2:2 Accumulated	Unit1:4:7 Current	Unit1:4:7 Accumulated	Total Current	Total Accumulated
2	Transmitted Bit Rate (bit/s)	5,117,440bit/s	5,117,078bit/s	512bit/s	12bit/s	-	-
3	Transmitted Bit Rate (%)	0.05%	0.05%	0.00%	0.00%	-	-
4	Transmitted Byte	639,68	26,225,024	64	64	639,744	26,225,088
5	Transmitted Frame	9995	409,766	1	1	9,996	409,767
6	**Error**	**99,95%**					
7	Transmitted Test Frame	0	0	0	0	0	0
8	Received Bit Rate (bit/s)	0bit/s	0bit/s	5,118,512bit/s	5,116,107bit/s	-	-
9	Received Bit Rate (Mbit/s) 1	0Mbit/s	0Mbit/s	-	-	-	-
10	Received Bit Rate (Mbit/s) 2	0Mbit/s	0Mbit/s	-	-	-	-
11	Received Bit Rate (Mbit/s) Other	0Mbit/s	0Mbit/s	-	-	-	-
12	Received Bit Rate (%)	0.00%	0.00%	0.51%	0.51%	-	-
13	Received Rate (%)	0.00%	0.00%	0.67%	0.67%	-	-
14	Received Rate (%) 1	0.00%	0.00%	-	-	-	-
15	Received Rate (%) 2	0.00%	0.00%	-	-	-	-

Fig. 19: Anritsu lost 0.05% of frames at a sending rate of 10,000 f/s

	A	B	C	D	E	F	G
1	Name	Unit1:2:2 Current	Unit1:2:2 Accumulated	Unit1:4:7 Current	Unit1:4:7 Accumulated	Total Current	Total Accumulated
2	Transmitted Bit Rate (bit/s)	5,438,749,152bit/s	3,208,679,599bit/s	324,874,752bit/s	191,665,125bit/s	-	-
3	Transmitted Bit Rate (%)	54.39%	32.09%	32.49%	19.17%	-	-
4	Transmitted Byte	679843644	7620614048	40609344	455204672	720452988	8075818720
5	Transmitted Frame	453229	5080410	634521	7112573	1087750	12192983
6	**Error**	**45,32%**		**63,45%**			
7	Transmitted Test Frame	453229	5080410	0	0	453229	5080410
8	Received Bit Rate (bit/s)	324,875,264bit/s	191,664,236bit/s	986,846,824bit/s	582,204,097bit/s	-	-
9	Received Bit Rate (Mbit/s) 1	0Mbit/s	0Mbit/s	-	-	-	-
10	Received Bit Rate (Mbit/s) 2	0Mbit/s	0Mbit/s	-	-	-	-
11	Received Bit Rate (Mbit/s) Other	324.875Mbit/s	191.664Mbit/s	-	-	-	-
12	Received Bit Rate (%)	3.25%	1.92%	98.68%	58.22%	-	-
13	Received Rate (%)	4.26%	2.52%	100.00%	59.00%	-	-

Fig. 20: Anritsu lost more than 50% of frames at a sending rate of 1,000,000 f/s

APPENDIX 3

```
set routing-options fate-sharing group FATE_TST cost 65535
set routing-options fate-sharing group FATE_TST use-for-post-convergence-lfa
set routing-options fate-sharing group FATE_TST from *.*.*.58 to *.*.*.59
set routing-options fate-sharing group FATE_TST from *.*.*.60 to *.*.*.61
set routing-options fate-sharing group FATE_TST from *.*.*.62 to *.*.*.63
set groups FATE_TST protocols isis interface <*> level 2 post-convergence-
    ↪ lfa fate-sharing-protection
set protocols isis interface ge-2/3/0.0 level 2 post-convergence-lfa apply-
    ↪ groups FATE_TST
set protocols isis interface ge-2/3/0.0 level 2 post-convergence-lfa fate-
    ↪ sharing-protection
set protocols isis interface ge-2/3/1.0 level 2 post-convergence-lfa apply-
    ↪ groups FATE_TST
set protocols isis interface ge-2/3/1.0 level 2 post-convergence-lfa fate-
    ↪ sharing-protection
```

```
set protocols isis interface ge-2/3/2.0 level 2 post-convergence-lfa apply-
    ↪ groups FATE_TST
set protocols isis interface ge-2/3/2.0 level 2 post-convergence-lfa fate-
    ↪ sharing-protection
```

Listing 1: Fate-sharing configuration

APPENDIX 4

```
set protocols isis backup-spf-options use-post-convergence-lfa
set groups ISIS_TI_LFA
set groups ISIS_TI_LFA protocols isis interface <*> level 2 post-convergence
    ↪ -lfa node-protection
set protocols isis apply-groups ISIS_TI_LFA
```

Listing 2: Fate-sharing configuration

APPENDIX 5

```
#!/usr/bin/python
import networkx as nx
from itertools import islice
##create 2 different metrics
metric1=5
metric2=10
##create a MultiGraph
MG=nx.MultiGraph()
## build the network of core routers in the surfnet network
## by adding edges to the multigraph
MG.add_weighted_edges_from([(0,1, metric1), (0,2, metric1), (0,2,metric2),
    ↪ (0,3,metric1), (1,2,metric2), (2,4, metric1), (2,3,metric2), (3,4,
    ↪ metric1), (4,5, metric1), (5,6,metric1), (6,8,metric1), (3,7, metric1)
    ↪ ,(7,5, metric1), (7,8, metric1), (8,6, metric1)])
##create a Graph
GG=nx.Graph()
##put all weighted edges in the Graph
for n, nbrs in MG.adjacency():
   for nbr,edict in nbrs.items():
      minvalue=([d['weight'] for d in edict.values()])
      GG.add_edge(n,nbr, weight = minvalue)


## put all routes in a list and sort the routes based on the weight

def k_shortest_paths(GG, source, target, k, weight=None):
   return list(islice(nx.shortest_simple_paths(GG, source, target, weight=
      ↪ weight), k))


#set source router on 0
SR = 0
#counter for total_paths on 0
total_paths = 0
#Get all paths from SR to DR
while (SR < 9):
   #set destination router to 0
   DR=0
   while (DR < 9):
```

```
      ##check if source router is equal to destination router,
      ##if so go to next destination router
      if SR == DR:
         DR = DR + 1
      else:
         ## set variable for primary and backup path
         primarypathcost=0
         secondairypathcost=0
         ## get the first 100 shortest paths
         for path in k_shortest_paths(GG, SR, DR, 100):
            cost=0
            totalcost=0
            # calculate the path cost
            while (cost+1 < len(path)):
               n1=(path[cost])
               cost=cost+1
               n2=(path[cost])
               totalcost=totalcost+(GG[n1][n2]['weight'][0])
            ##check if there is already a primairy path is calculated
            if primarypathcost == totalcost:
               print(path),(totalcost)
               #increase the counter of found paths
               total_paths = total_paths+1
            elif primarypathcost == 0:
               primarypathcost = totalcost
               print(path),(totalcost)
               #increase the counter of found paths
               total_paths = total_paths+1
            ##check if there is pathcost is higher than the primairy path
            elif totalcost > primarypathcost:
               if secondairypathcost == totalcost:
                  print(path),(totalcost)
                  #increase the counter of found paths
                  total_paths = total_paths+1
               elif secondairypathcost == 0:
                  secondairypathcost = totalcost
                  print(path),(totalcost)
                  #increase the counter of found paths
                  total_paths = total_paths+1
               else:
                  break
            else:
               break
         #Go to next Destination router
         DR = DR + 1
   #Go to next source router
   SR = SR + 1
print('total_paths'),(total_paths)
```

Listing 3: Python Path Cost Calculator