Towards Optimizing Fast Rerouting under Multiple Failures

Stephanie Althoff[1] and Klaus-Tycho Foerster[1]

[1]Computer Science, TU Dortmund, Dortmund, Germany, ✉ firstname.lastname@tu-dortmund.de

Abstract

With the increasing size and relevance of computer networks, we are in need of mechanisms to reduce the impact of the rising number of failures within the network. In this paper, we present an evaluation of the FRR extension Shortcut and show how it can reduce hop count in fault scenarios while maintaining the resilience of the underlying FRR mechanism.

1 Introduction and System Design

In recent decades, the Internet and its use cases have been strongly growing. However, with expanding scale, the network becomes more vulnerable to failures [1, 3]. With the increasing use of the internet through time-critical applications, there is a great demand for resilient networks that can offer swift reactions against multiple failures and attacks. One way to achieve this, are fast reroute (FRR) mechanisms which offer quick responses to outages. In this work, we evaluate how the FRR extension ShortCut [2] can reduce hop count and therefore, minimize latency by creating loop-free paths in fault scenarios.

Our network is modeled as a graph $G = (V, E)$ with n nodes V and m undirected links E. In a failure-free scenario, a packet of flow f is routed from a source node $s = s(f) \in V$ to a destination node $t = t(f) \in V$ along the shortest edge-disjoint path (EDP) connecting those nodes. The forwarding decision at a node v on which outport v^o of v to use is deterministic and depends only on the following information: source s, destination t and inport of the packet v^i at v.

The underlying FRR mechanism used for this work is called SquareOne (SQ1) [1]. SQ1 leverages edge-disjoint paths (EDPs) which are computed between every pair of nodes within the network. The paths are chosen regarding their length in descending order. If the network encounters a failure on the path of highest priority, packets are sent back to s and rerouted over the EDP with next highest priority. In general, if there is a fail-free EDP left between two nodes, the router reroutes to the next available path. However, this results in loops on the paths with failures which leads to higher latency for packets as well as congestion. To mitigate unnecessary loops, we introduce ShortCut [2] to our setup.

ShortCut operates at every source node $s \in V$. If the forwarding outport v_h^o of a packet does not correspond to the first item in the priority list $v_j^o, \ldots, v_k^o$ of inport v^i for flow f, ShortCut removes all front outports from the priority list until the forwarding outport v_h^o is the first item. In essence, ShortCut observes the use of a lower priority outport and makes it the new highest priority outport for all corresponding inports, removing all outdated outports with former higher priority. Therefore, preventing loops after detection, saving latency and avoiding congestion. ShortCut strictly operates locally at each node, no control plane messages or exchange between the nodes are needed.

We can show that in our setting the following holds, here omitted due to space constraints:

Theorem. When the FRR scheme SQ1 maintains reachability under multiple link failures, then Shortcut with SQ1 maintains reachability and changes the route to a loop-free FRR sub-path.

Corollary. After a link failure, all ShortCut route change actions trigger within one end-to-end delay. For every additional failure, ShortCut route change actions trigger within an additional end-to-end delay.

2 Evaluation

In this section we aim to evaluate our ShortCut-SQ1-extension regarding its reachability and hop count in comparison to the standalone FRR mechanism under multiple link failures. We implement our experiments using Python 3.11 and the associated package NetworkX 2.8.8 which allows working with complex networks. The experiments are divided between different graph types and failure models.

The first graph type is a random regular graph with a connectivity of k=8. The size of the graph is 100 nodes. For each graph, we iterate through all combinations of source and destination nodes where ShortCut is always activated on the respective source node. For statistical reasons, each measurement

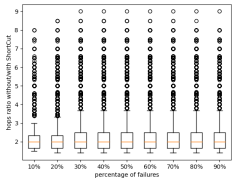

Figure 1: Random regular graph with 100 nodes, random failures

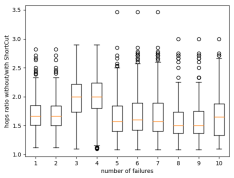

Figure 2: Cogentco ($n = 197$), random failures

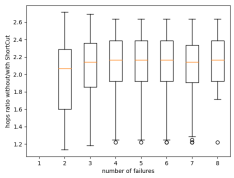

Figure 3: Oteglobe ($n = 93$), random failures

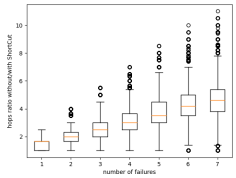

Figure 4: Random regular graph with 100 nodes, targeted failures

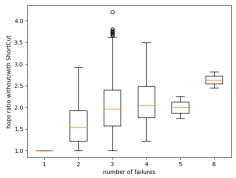

Figure 5: Cogentco ($n = 197$), targeted failures

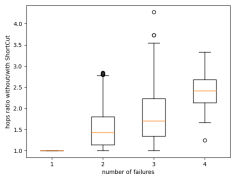

Figure 6: Oteglobe ($n = 93$), targeted failures

is repeated for a total of 100 random generated graphs. For the second graph type we chose two real topologies, namely Cogentco and Oteglobe from `http://www.topology-zoo.org`.

In the random failures model, failures are placed randomly and simultaneously within the network. We vary the number of failures from 1 to 10 failures for real topologies and 10% to 90% on the network's number of edges for generated graphs or until there is no more feasibility of routing. When using SQ1, there is a definite number of EDPs between a source and destination node. Instead of throwing random failures into the network, we explicitly create failures on those paths. With SQ1, EDPs are sorted in ascending order in terms of lengths. The first failure is placed randomly on the shortest path. The second failure is placed sequentially on a random location of the second shortest path and so on until there is no fail-free EDP left. The route between the two nodes becomes unrouteable. The number of feasible failures, so that routing is still feasible, corresponds to the number of EDPs minus 1. We call this the targeted failures model.

Our results are displayed as boxplots of the ratio between the hops count of only the FRR mechanism and with use of Shortcut. Figures 1 to 3 show the results with a difference in the hop count ratio for random failures. ShortCut has a positive impact on the hop count ratio with a median around 1.5 to 2.0. In Figures 4 to 6, you can see the results for the targeted failures model. It is valid for all three networks that with an increasing number of failures, the ratios go up. This means that the use of ShortCut reduces the hop count and its effect is stronger with higher failure rates.

3 Conclusion and Outlook

This poster paper shows that fast rerouting can be optimized in a loop-free manner using ShortCut, in order to improve e.g. latency and throughput. We showed that ShortCut can be deployed to an existing FRR mechanism and never affects its performance negatively. In fact, in many cases, especially with multiple failures, ShortCut improves performance by reducing the hop count and, therefore, lowering the latency. Moving forward, we want to further evaluate ShortCut's impact on FRR mechanisms under multiple failures. We achieve this by deploying time-sensitive simulation and small-scale experiments on hardware using a variation of FRR algorithms. Moreover, we want to develop further guarantees and improvements for ShortCut's performance.

References

[1] Klaus-Tycho Foerster, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. CASA: Congestion and Stretch Aware Static Fast Rerouting. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pages 469–477, Paris, France, April 2019. IEEE Press.

[2] Apoorv Shukla and Klaus-Tycho Foerster. Shortcutting Fast Failover Routes in the Data Plane. In *Proceedings of the Symposium on Architectures for Networking and Communications Systems*, ANCS '21, pages 15–22, New York, NY, USA, January 2022. Association for Computing Machinery.

[3] World Economic Forum. Global cybersecurity outlook 2025. Report, World Economic Forum, Geneva, Switzerland, 2025. Accessed: 2026-02-06.