Loop-Free Convergence using Ordered FIB Updates: Analysis and Routing Optimization

David Hock*, Matthias Hartmann*, Tim Neubert*, Michael Menth‡
*University of Würzburg, Institute of Computer Science, Würzburg, Germany
Email: {hartmann,hock,tim.neubert}@informatik.uni-wuerzburg.de
‡University of Tübingen, Department of Computer Science, Tübingen, Germany
Email: menth@informatik.uni-tuebingen.de

***Abstract*—Intradomain IP routing protocols like OSPF or IS-IS are robust against failures. New fast reroute mechanisms can minimize packet loss directly after an outage by temporarily detouring packets around the failed network element. This buys time for a loop-free convergence, which brings the topology to a new stable path layout without causing temporary micro-loops.**

We analyze the link utilizations during this loop-free convergence phase using the ordered FIB updates (OFIB) mechanism. We show that depending on the router update order, OFIB can temporarily increase the utilization on certain links in the network. To minimize the temporary load increase, we present a heuristic link cost optimization that minimizes the link utilizations both during failure-free routing and all phases of a failure recovery process. As OFIB does not define a unique global update order but provides only local constraints, it is difficult to calculate the highest possible link utilizations. We introduce a tight upper bound to the maximum link utilization, independent of the actual update order. It can quickly be calculated and allows us to perform link cost optimization including the loop-free convergence phase. We show that this results in a routing configuration that avoids additional overload during the OFIB phase without impairing normal routing performance.

***Index Terms*—loop-free convergence; ordered FIB updates; OFIB; fast reroute; not-via addresses; routing optimization; maximum link utilization; traffic distribution**

I. Introduction

Link state routing protocols combined with currently standardized fast reroute and loop-free convergence mechanisms are very robust to changes in the network topology, e.g., due to link or router failures. Information about the changed topology is distributed among the routers, the routes are re-calculated and the routing converges to a new stable state. As this update can take up to several seconds, fast reroute mechanisms provide precalculated backup paths so that each router can instantaneously shift the traffic to other paths when detecting a failure. During the following convergence phase, micro-loops might appear temporarily, when routers update their forwarding table (FIB) in an unfavorable order and packets loop between routers with different new and old routing information. Special loop-free convergence mechanisms avoid these loops by defining certain constraints for the FIB update order of the routers. The combination of these mechanisms generates a resilient IP routing that ensures almost uninterrupted communication also during outages in the network.

Nevertheless, while these mechanisms are used, the utilization of certain links may significantly increase. Some links might even experience overload and subsequent packet loss. We investigate this behavior, using not-via addresses [1] as fast reroute mechanism and ordered FIB updates (OFIB) [2] as loop-free convergence mechanism. We show that OFIB often temporarily increases the utilization on certain links in the network, possibly causing overload on these links. We quantify the impact of OFIB on the maximum link utilization for several well-known example topologies. We show that the temporary utilization increase can be significant and is an non-negligible issue. It is especially important to consider this for resilient routing optimization trying to reduce the maximum link utilization also during certain failure cases.

The OFIB concept does not provide a fixed update order but only provides certain constraints. The number of OFIB-conform update orders is exponential with regard to the number of routers in the network, so it is computationally not feasible to analyze all possible update orders. We first try to find the maximum link utilization during the OFIB phase by simulating a number of different valid update orders. Then, we introduce an algorithm that finds an order-independent upper bound to this maximum link utilization. The upper bound is tight, can be computed quickly, and allows us to extend our heuristic routing optimization framework [3]–[5] to consider the OFIB process. To the best of our knowledge, we provide the first resilient IP routing optimization that considers not only the failure-free, fast reroute and failure case but also the loop-free convergence phase.

The paper is structured as follows. Section II explains failure handling in link state routing, summarizes its different phases, and explains ordered FIB updates. An overview of related work is given in Section III. In Section IV, we illustrate different types of temporary utilization increase. Section V analyzes and quantifies the temporary utilization during the loop-free convergence phase. An upper bound algorithm is introduced to efficiently analyze OFIB and consider it during optimization. Section VI presents and discusses the results of our routing optimization. Section VII concludes the paper.

‡ The work was done while the author was at the University of Würzburg.
Funded by Deutsche Forschungsgemeinschaft (DFG) under grant TR257/23-2. The authors alone are responsible for the content of the paper.

II. Failure Handling in Link State Routing

In this section, we outline the different phases of failure handling in link state routing, including fast reroute and loop-free convergence.

A. General Link State Routing and Re-Convergence

In typical intradomain IP networks routers exchange information about the topology and the costs of each link. Each router calculates least-cost routes to all destinations in the network and creates a forwarding table (FIB) that stores the next hop for each destination.

After a failure in the network, information about the changed topology is distributed in the network. Each router recalculates its least-cost paths omitting the failed elements. The network re-converges to a regular routing and communication can continue as long as the network is physically connected. This makes IP routing very robust to failures.

B. Fast Reroute

During the re-convergence process, which can take several seconds, traffic is lost. Fast reroute mechanisms can avoid this. Routers precalculate backup paths for certain failures and immediately switch to an alternative route after detecting a failure. In this paper we use the fast reroute mechanism *not-via addresses* [1]. It precalculates tunnels around the failure to the next next hop $NNHOP$ (i.e., the hop after the next hop). We use not-via addresses as they provide 100% failure protection coverage for all single link and router failures. Other fast reroute mechanisms fullfiling this criteria could be used as well.

C. Phases of the Failure Handling Procedure

The entire failure handling procedure in link state routing protocols like IS-IS or OSPF can be divided in five different phases, as illustrated in Figure 1.

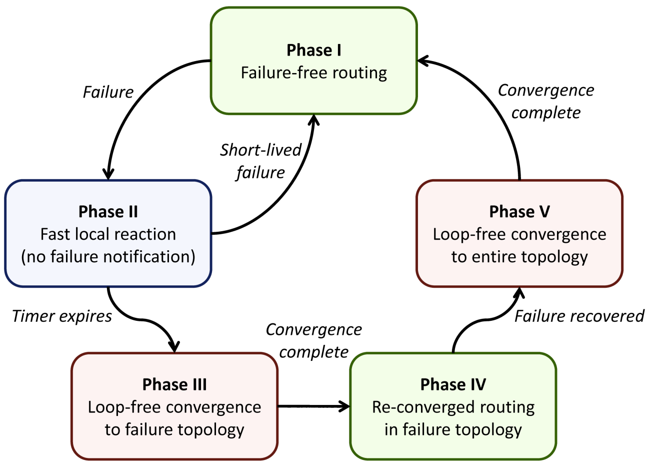


Fig. 1. Phases of an IGP failure handling procedure.

The normal operation state of any network is the *failure-free routing* (Phase I). All links and routers are operating normally and traffic is routed according to least-cost paths.

After a failure, the network enters the *fast local reaction phase* (Phase II). Most of the traffic is still routed according to least-cost paths. The traffic affected by the failed components is sent on backup paths using a fast reroute technique. To avoid unnecessary network-wide re-convergence during short-term failures, the failure is not broadcast in the entire network.

The use of fast reroute techniques increases the load on backup paths and leads to longer average path lengths. Furthermore, most fast reroute mechanisms are only designed to handle a single failure and subsequent failures would cause packet loss. Thus, after a preconfigured timer expires, a failure notification is distributed in the network and the network enters the IP re-convergence phase. This process (Phase III) can take several seconds. During this time, micro-loops might appear, where packets loop in between routers with different views of the network. When, for example, router A in Figure 2 updates its forwarding table after the failure of link $A \leftrightarrow B$, it uses C as next-hop towards B. But if router C has not yet updated, it sends these packets back to node A and a micro-loop is created. Without IP fast reroute mechanisms, this was acceptable because packets would be lost anyway until all routers have updated. With fast reroute, almost no packets are lost initially. An uncontrolled re-convergence would then render the fast reroute detour useless, when the failure detecting router updates its FIB and sends packets to the new next hop. The subsequently occurring micro-loops could, in addition, lead to overload on other links and possibly impair otherwise unaffected traffic. Therefore, loop-free convergence has to be guaranteed. While the routers re-converge loop-free to the new routing, fast reroute mechanisms continue to assure that no traffic is lost. In the re-converged routing phase (Phase IV), all traffic is routed according to least-cost paths in the failure topology. When the failure can be repaired, the routing converges back to the original failure-free state, again using a loop-free convergence mechanism (Phase V).

D. Ordered FIB Updates

In this paper, we use the ordered FIB updates (OFIB) [2], [6] loop-free convergence mechanism. OFIB assures loop-free convergence by imposing certain rules on the update order of the routers in the network. We explain the general idea of OFIB, using the example in Figure 2 where a link fails and then reappears again. We call these events link-down and link-up event, respectively.

1) Terminology: First, we explain the OFIB terminology. A failure of the bidirectional link $A \leftrightarrow B$ can be regarded as two unidirectional failures of links $A \rightarrow B$ and $B \rightarrow A$.

The *reverse shortest path tree* $rSPT(B)$ of router B is formed by the shortest paths of all routers towards the destination B. In Figure 2, we only show $rSPT_A(B)$, the *reverse shortest path tree regarding a link* $A \rightarrow B$. It is the subtree of $rSPT(B)$ that is attached to the router A. Thus, it is formed by all routers whose shortest paths to B include the link $A \rightarrow B$. The $rSPT_B(A)$ is constructed likewise with all routers whose shortest paths to node A include $B \rightarrow A$.

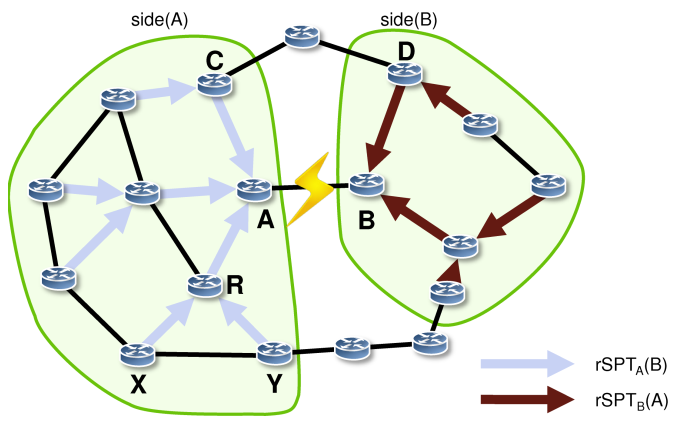


Fig. 2. Reverse shortest path tree towards a single link failure.

Each shortest path can contain at most one direction of a link $A \leftrightarrow B$, so $rSPT_A(B)$ and $rSPT_B(A)$ are disjoint.

Sides $side(A)$ and $side(B)$ of a link failure $A \leftrightarrow B$. With regard to the failure of a link $A \leftrightarrow B$, a network can logically be split into two sides. All routers that form $rSPT_A(B)$ and the links connecting them are located on $side(A)$ of the failure, while all routers of $rSPT_B(A)$ are located on $site(B)$. Routers that do not use the link $A \leftrightarrow B$ are not assigned to any of both sides.

2) Link-Down Event $A \leftrightarrow B$: After a link-down event of link $A \leftrightarrow B$, micro-loops can appear only if an already updated router sents packets to a router that has not updated yet. To assure loop-free convergence on $side(A)$, a router R has to postpone its update until all other routers that send traffic via R and $A \rightarrow B$ have updated their FIBs first. Hence, the updates are conducted starting from the leaves of $rSPT_A(B)$, so that the routers farthest from the failure update first, the ones next to the failure update last. This prevents micro-loops during the re-convergence process [6].

3) Link-Up Event $A \leftrightarrow B$: Link-up events are handled likewise. Similarly to the link-down event, the reverse shortest path trees $rSPT_A(B)$ and $rSPT_B(A)$ are considered[1]. In this case, the updates in the rSPTs are conducted starting from the roots. Router R on $rSPT_A(B)$ delays its FIB update until all predecessors on $rSPT_A(B)$, i.e., all routers that R uses to transmit traffic via link $A \rightarrow B$, have updated their FIBs. Again, loop-free convergence is assured.

4) Update Order: OFIB is based on certain update order constraints, which can be achieved with two mechanisms [2].

The first technique is based on timers. Each router calculates its so-called rank in the rSPT and, depending on that, a certain waiting time before starting the update. Figure 3 depicts the ranks for all routers of an arbitrary $rSPT_B(A)$. The two numbers assigned to each of the routers indicate the rank for a link-down and a link-up event. In case of a link-down event, the leaves of the rSPT (F, G, D, I, and J) are the first to update, and therefore, have rank 1. After the configured maximum update time, all routers with the next rank (C and H) start their update. This process is continued until all routers including the root of the rSPT have updated. The update order in case of a link-up event works vice versa. The first router to update is the root of the rSPT, router B. It is followed by the routers of the next rank (C, D, and E), and so forth. To make sure that all update order constraints are fulfilled even if some router's update takes longer, the waiting times for the timer-based update have to be chosen sufficiently large.

[1]These are the same rSPTs as before where link $A \leftrightarrow B$ is working.

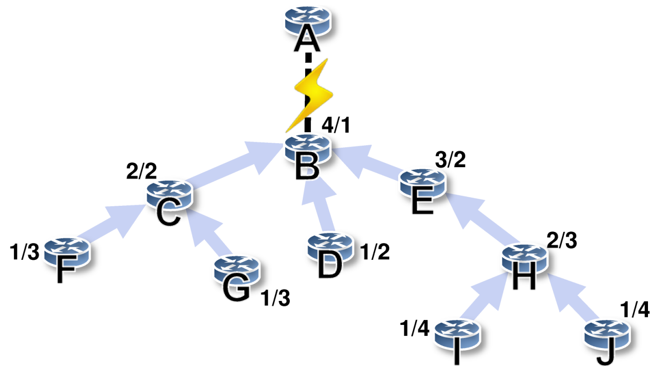


Fig. 3. Timer-based OFIB. The two numbers assigned to each router indicate the update rank for a link-down and link-up event subsequently.

As the timer technique is slow, a message-based technique is proposed. In this case, each router R has a waiting list with other routers that still have to update before R and a notification list with routers that are waiting for R's update. As soon as R's waiting list is empty it updates and then, notifies all routers in the notification list. For example router C in Figure 3 waits for the updates of F and G, then updates, and finally, notifies B about its update. Using this technique, all routers can directly update as soon as the constraints are fulfilled, which significantly accelerates the OFIB process.

III. Related Work

In previous work [4], [5], we provided an extensive overview of related work in the area of link cost optimization and IP fast reroute. To the best of our knowledge, there is only few related work concerning loop-free convergence and oFIB updates. We briefly describe it in the following.

The RFC5715 [7], based on the analysis of Zinin [8], discusses the causes for micro-loops in general, gives an overview on counter measures to micro-loops and discusses how the number of micro-loops can be minimized. Francois et al. [6], [9] show that micro-loops may occur during the convergence of link state routing protocols depending on the update order. Furthermore, they introduce the OFIB concept and thereby show that it is possible to define update orders that effectively avoid micro loops. Finally, they show by simulations that sub-second loop-free convergence is possible on a large Tier-1 ISP network. Link utilizations during OFIB are not considered in these papers. Fu, Shi et al. [10], [11] address the problem of temporary load increase during the loop-free convergence phase. They propose to tackle this issue by calculating special update orders that reduce the load increase. The basic idea is to always reroute the flow that causes the least overload. Shi et al. [11] extend the idea of Fu et al. [10] by assuming that the nodes do not need to update their entire forwarding table en block but that they can do partial updates of flows one by one. This heuristic requires several iterations that leads to the proposed algorithm being slower than the one of Fu et al. [10]. The methods of both papers have been tested in example networks for an number of failure scenarios. Both papers revealed that modifying the update orders brings no guarantee to avoid temporary load increase during the loop-free convergence phase.

In this paper, we do not try to improve the update order but we try to optimize administrative link costs in such a way that the temporary load increase is avoided independent of any particular update order.

IV. Temporary Utilization Increase Caused by OFIB Orders

OFIB solves the problem of micro-loops. During the loop-free convergence phase the step-wise updates can lead to a temporary utilization increase on certain links in the network. In this section, we illustrate two different types of temporary utilization increase.

A. Looking at One Side of the Failure

The first type of temporary utilization increase appears on a single side of the failure, e.g., $side(A)$ of link $A \leftrightarrow B$. It is caused by two or more nodes that can update independently of each other because they are in different subtrees of $rSPT_A(B)$, e.g., routers X and Y in Figure 2.

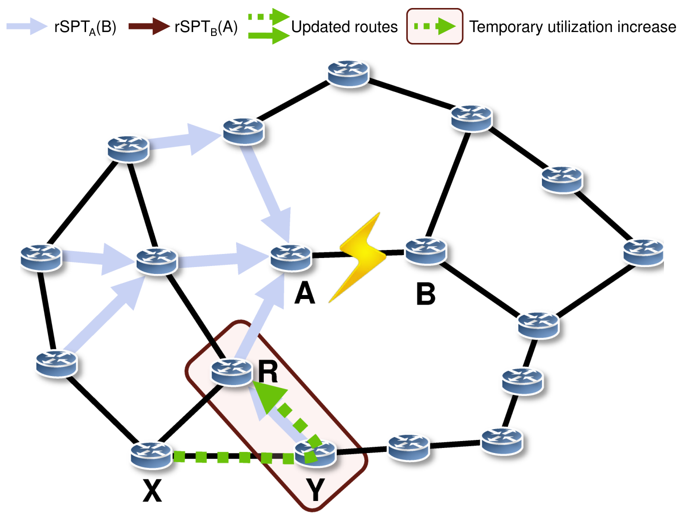
Assuming that synchronous updates are technically not feasible, there are two possible update orders for X and Y that comply with the OFIB rules. Both orders are illustrated in Figure 4 and will be discussed in the following.

If X updates its FIB first, see Figure 4(a), Y sends the traffic originated in X via the old routing tree until it updates its own FIB. The network experiences a temporary utilization increase on link $Y \rightarrow R$ that disappears after the update of Y, see Figure 4(c). This situation is avoided when Y updates first, followed by X, see Figure 4(b).

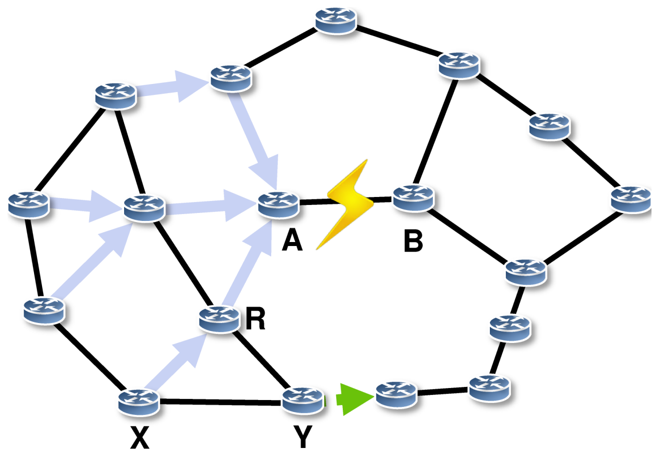
The example shows that OFIB may lead to unpredictable utilization increase when independent routers incidentally update in a correct but disadvantageous order.

B. Interference between Different Failure Sides

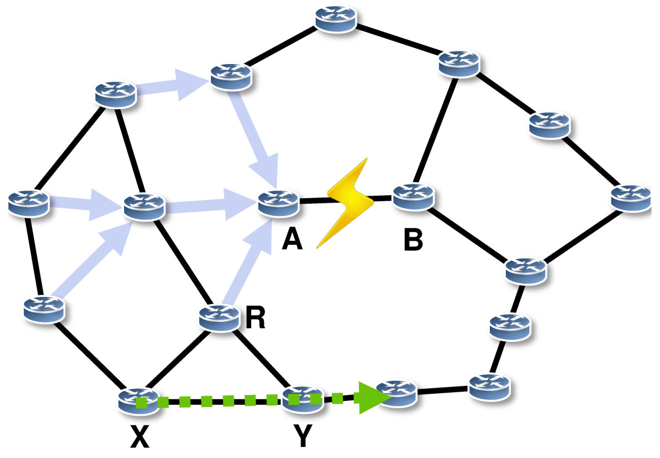
The second type of temporary utilization increase appears by an interference of different sides of a failure. The update of a router on $side(A)$ in Figure 2 might cause temporary utilization increase on links of $side(B)$. Routers C and D can update independently as they are part of different rSPTs $rSPT_A(B)$ and $rSPT_B(A)$, respectively. Again, there are two possible update orders. The effects of both are displayed in Figure 5. In both cases the network experiences a temporary utilization increase on a link on the other side of the updating router, i.e., an update on $side(A)$ influences a link on $side(B)$



(a) X updates before Y.

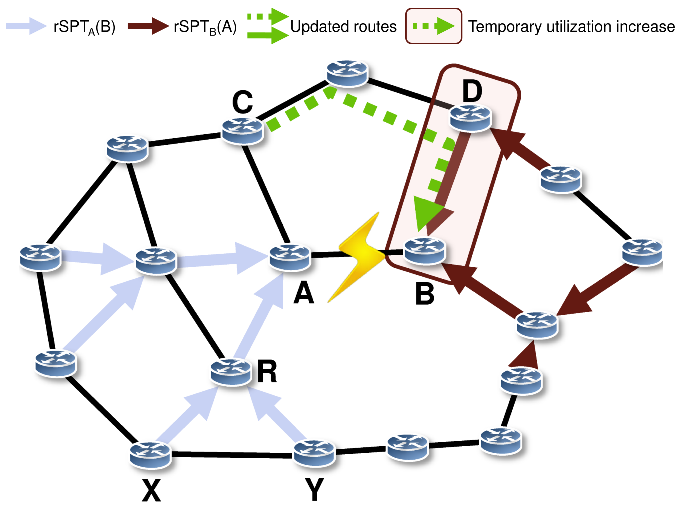


(b) Y updates before X.

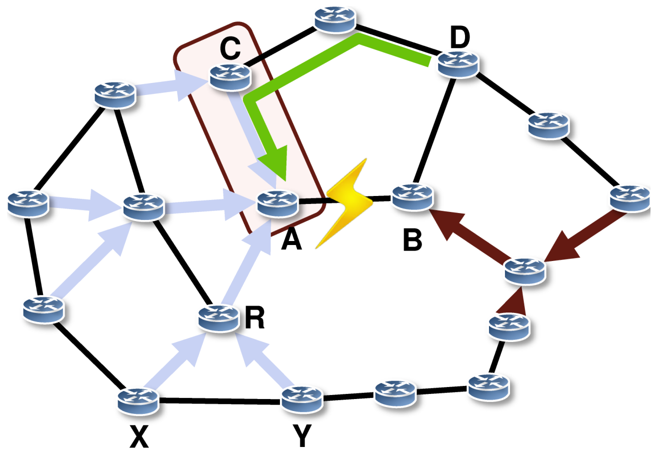


(c) After both updates.

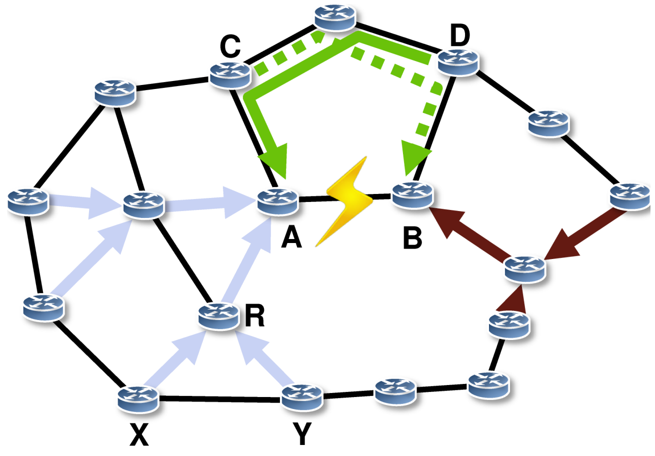
Fig. 4. Example for temporary utilization increase on single network side: When router X updates before router Y, see Figure 4(a), the network experiences a temporary utilization increase on link $Y \rightarrow R$ that disappears after the update of Y, see Figure 4(c),. This situation is avoided when Y updates first, see Figure 4(b).



(a) C updates before D.



(b) D updates before C.



(c) After both updates.

Fig. 5. Example for interference between different failure sides: When routers of $rSPT_A(B)$ update before routers of $rSPT_B(A)$, temporary utilization increase appears on side(B) and vice versa. Changing the update order might not avoid this problem completely but shift the temporary utilization increase to different links.

and vice versa. Note that there is additional load on some links caused by the fast reroute detour that is not shown in the figure.

If C updates first as in Figure 5(a), additional packets are routed over the link $D \rightarrow B$. Router D has not updated yet and also sends packets over the same link. After the update of D, the utilization of link $D \rightarrow B$ decreases again because the new path from D to A does not include router B anymore, as depicted in Figure 5(c). If, on the other hand, D updates first as in Figure 5(b), the same temporary utilization increase appears on link $C \rightarrow A$.

The example shows that updating routers on one side of the failure may cause a temporary utilization increase on the other side. Not even regarding the necessary signalling overhead, it is hard or even impossible to automatically determine which side of the failure should start updating first.

V. Analysis of the Maximum Link Utilization during the OFIB Phase

The previous section showed that OFIB can lead to a temporary utilization increase. In this section, we use several well-known network topologies to quantify the impact of the temporary utilization increase during the loop-free convergence phase on the maximum link utilization, i.e., the utilization of the most loaded link in the network. First, we explain how we calculate the maximum link utilization during OFIB considering different possible update orders. Afterwards, we present our numerical results. To ease the analysis and handling of the temporary utilization increase, we present an update-order-independent algorithm that gives tight upper bounds to the maximum link utilization before, after, and during the loop-free convergence phase.

A. Calculating the Maximum Link Utilization considering Different Update Orders

During the OFIB phase, many subsequent update steps are performed. To obtain the maximum link utilization including any temporary utilization increase during OFIB, we must calculate the link utilizations after each OFIB update step. The OFIB concept does not provide a fixed update order but only constraints that any possible update order has to fulfill. We illustrate this using Figure 3. The $rSPT_B(A)$ can be divided into three subtrees (C, F, G), (D), and (E, H, I, J). Each of these subtrees can conduct the OFIB updates independently of the others. The root B has to wait until all subtrees have finished before starting its update. Likewise, each node in a subtree has to wait until its children in the tree have updated. The duration of each update depends on many factors and is, in general, unknown. This leads to many possible OFIB orders[2].

The number of OFIB-conform update orders is exponential with regard to the number of routers in the network. Thus, it is computationally not feasible to analyze all possible update orders and to obtain the worst case maximum link utilization. To estimate the maximum link utilization, we create a set

[2]Similar to the different subtrees of router B, also the two sides of the failure, $rSPT_B(A)$ and $rSPT_A(B)$, can update independently, which further increases the number of update orders

of 1000 different OFIB-conform update orders by varying individual update durations. This technique does not lead to the actual theoretical maximum link utilization. However, our experiments have shown that the evaluation of 1000 update orders already illustrates the utilization increase during the OFIB phase at acceptable computational effort. These orders are evaluate step by step and we choose the highest occurring maximum link utilization value as result of our simulation.

B. Numerical Results

In the following, we provide information about the networks and link cost settings under study. Then, we analyze the influence of the temporary utilization increase during OFIB on the maximum link utilization.

1) Networks and Routings under Study: We analyze several widely used research topologies as well as some of the well-known Rocketfuel topologies [12]: Abilene (AB), AT&T (AT), Cost239 (CO), EBone (EB), Exodus (EX), Geant (GE), Labnet (LA), Nobel, (NO), Sprintlink (SP), and Tiscali (TI). All networks have been reduced to their two-connected core, i.e., the considered parts of the networks are still physically connected after any single link or router failure. The traffic matrices are created based on a gravity modell and scaled so that the maximum link utilization during all single link failures is exactly 100%. For this purpose, the administrative link costs in the unoptimized networks were all set to 1.

The link cost settings that are analyzed in the following have been obtained by IP link cost optimization. In [3], we proposed a heuristic link cost optimization and extended it to consider fast reroute in [5]. We used this threshold accepting heuristic to obtain routing configurations for each network with low maximal link utilizations. The optimization took into account the failure free routing as well as the fast reroute not-via detours and the reconverged routing after failures. The OFIB phases were not considered during the optimization. In almost all networks, the link utilization are highest during the not-via fast reroute phase. Thus, we refer to this maximum link utilization as *fast reroute* utilizations. In this paper, we describe only the results for single link failure scenarios. We also evaluated router failures and obtained similar results.

2) Numerical Results: We use the optimized link cost settings and evaluate the maximum link utilizations also during the OFIB phase. Figure 6 shows the fast reroute utilization values compared to the highest simulated OFIB utilizations in all networks under study. The effect of a possible temporary utilization increase during OFIB on the maximum link utilization is different depending on the network and the link cost setting. In some of the networks, OFIB leads to no or little additional maximum link utilization. However, in AB, EX, GE, NO, and TI the increase is quite significant.

The analysis shows that OFIB can have a big impact on the maximum link utilization and therefore should be considered during the link cost optimization process.

C. Algorithm to Obtain a Tight Upper Bound

In Section V-A, we proposed to analyze the maximum link utilization during OFIB with a simulation of several random

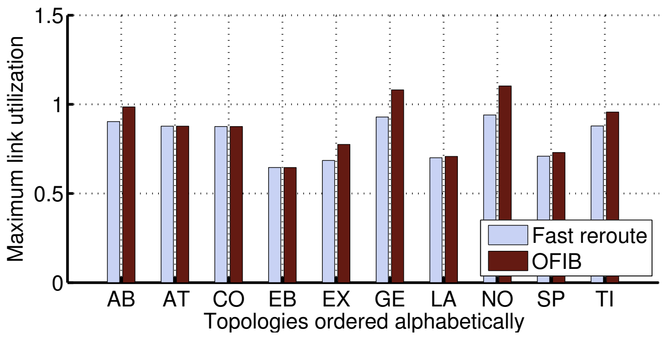


Fig. 6. Maximum link utilizations during fast reroute and OFIB phase.

update orders. This has two major drawbacks. First, the computation effort is large as several OFIB orders are regarded and need to be evaluated. Second, to effectively consider the OFIB maximum link utilization during routing optimization and to provide guarantees for the optimization results, the worst OFIB maximum link utilization has to be considered. However, there is no guarantee that the calculated values are even close to the worst OFIB maximum link utilization because only a small random subset of possible orders is analyzed.

To provide a computationally fast calculation of the OFIB maximum link utilization that guarantees the quality of the optimization, we propose an algorithm that provides update-order-independent upper bounds for the worst OFIB maximum link utilization. In the following, we first explain the algorithm and then show that the provided upper bounds are tight. Finally, we briefly discuss the computational effort of the proposed algorithm.

1) Algorithm description: The basic concept of the algorithm is quite simple: during the fast reroute and loop-free convergence phase, the routing changes and traffic flows can be routed on different links. An upper bound to the maximum utilization of a link can be obtained by summing up the size of every flow that could be routed over this link in any possible network phase: the failure-free and re-converged case, the fast reroute case, as well as every possible OFIB update order.

Summing up all utilizations caused by all flows to a single link provides an upper bound but not necessarily a value that can really occur during a particular update order. Still, in Section V-C2 we will show that the upper bounds are very tight for all considered networks.

To obtain all flows ever contributing to the utilization of a certain link the link utilizations caused by each flow are calculated separately. Repetition of this procedure for all flows that are affected by the failure leads to an upper bound of the total maximum link utilization. Figure 7 illustrates the algorithm to calculate the upper bound on the network discussed before. We consider the flow from X to Z in case of a failure of link $A \leftrightarrow B$. The figure shows a part of the original rSPT to Z, $rSPT(Z)$, in the failure-free case, and the new rSPT to Z in the failure-case $rSPT^*(Z)$, as well as the not-via backup path from the failure detecting router A to the next next hop $NNHOP$.

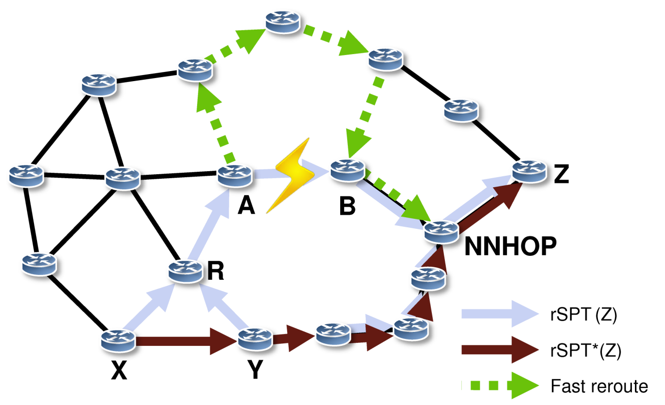


Fig. 7. A single flow spread over all links it could use in any routing phase.

The algorithm starts at the source of the considered flow, router X. Two cases have to be considered: X has already updated the FIB or it has not update the FIB yet. In the former case, the flow X to Z is routed on $rSPT(Z)$. All routers on the path to the failure A have not yet updated to fulfill the OFIB constraints since they must wait for the update of X first. The algorithm adds the demanded traffic rate of the regarded flow to every link's utilization on the old path towards the failure (from X to A), on every link of the fast reroute tunnel (from A to $NNHOP$), and on the subsequent links towards the destination (from $NNHOP$ to Z). In the second case, if X has updated its FIB already, the flow is sent to Y. Therefore, the algorithm adds the flow's traffic rate to link $X \rightarrow Y$.

At router Y, the same decisions are repeated again. If Y has not updated yet, the flow is sent on $rSPT(Z)$ via $Y \rightarrow R$. The algorithm has already added the flow to the link utilizations of the subsequent links before and therefore does not add them again. If Y has already updated, the flow follows the updated routing on $rSPT^*(Z)$ to the next router where the described decisions are again repeated. As soon as $rSPT(Z)$ and $rSPT^*(Z)$ merge, the algorithm can be terminated as all possible links the flow might be routed on have been considered.

This procedure ensures that the traffic rate of the flow from X to Z is added to the link utilizations of all links that can ever transport it during the OFIB phase, regardless of the specific update order.

2) Tightness of the Upper Bound: As mentioned before, the algorithm provides only an upper bound to the OFIB maximum link utilization. In the following, we show that this upper bound is mostly tight. To that end, we applied the upper bound algorithm to all link cost settings obtained during the previous optimizations. We evaluated the same link cost settings using the previously described simulation of 1000 different OFIB-conform update orders and took the worst found result. Then, we calculated the relative difference between the obtained upper bound value and the worst OFIB value obtained by the simulated OFIB orders. Figure 8 shows the complementary CDF (CCDF) of this relative difference values over all evaluated link cost settings. A relative difference of, for example 50% indicates that the obtained upper bound value is 150% of the worst found OFIB value, i.e., 50% worse than this value.

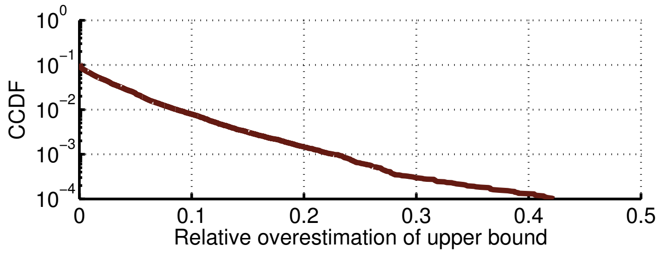


Fig. 8. Relative overestimation of the upper bound compared to the worst found OFIB value.

For about 90% of the investigated link cost settings, the highest simulated OFIB utilization is as high as the upper bound. Thus, for most link cost settings, the maximum link utilization of the upper bound algorithm represents a real value that can actually occur in the network. In more than 99% the relative difference between the upper bound and the worst found OFIB value is less than 10%. Even in the cases where these values differ, the upper bound might still represent a realistic utilization, because our simulation of OFIB uses only a limited number of update orders, and the actual worst case might not have been evaluated.

Our evaluations show that the algorithm provides a good upper bound which permits the use for routing optimization.

3) Computational effort to calculate the Upper Bound: The additional effort caused by the upper bound is almost neglectable for two reasons. First, during the optimization process our heuristic rejects solutions that are far from the current best value already in an earlier state. The OFIB upper bound is only computed for the few link cost settings that lead to not-via maximum link utilizations that are equally good or better than the current best value found so far. Second, for the evluation of our upper bound, we do not have to calculate any additional Dijkstra shortest path threes. The required original $rSPT$ and the new $rSPT^*$, see Section V-C, are already computed when considering the failure-free routing and the routing in the reconverged state. We just have to place the flows onto all possible links and recalculate the maximum link utilization.

VI. Optimization of the Maximum Link Utilization during the OFIB phase

Our previous analysis has shown that the maximum link utilization can increase significantly during the loop-free convergence phase. In this section, we show that this effect can be minimized by link cost optimization.

We extended our heuristic optimizer mentioned in Section V by implementing and integrating the upper bound algorithm presented for OFIB. This allow us to consider the loop-free convergence phases during the optimization process. We optimize the link cost settings as before to minimize the fast reroute link utilization. In addition, the OFIB utilization is optimized as a secondary goal. This way, we expect to find link cost settings of equal maximum link utilization in the previously analyzed scenarios and, in addition, reduce the temporary utilization increase during OFIB

To analyze the efficiency of our extended link cost optimization, we compare the fast reroute optimized link cost settings presented in Section V to link cost settings that have additionally been optimized for the OFIB loop-free convergence phases.

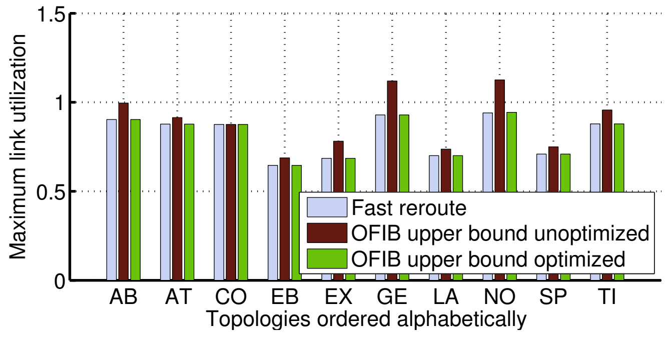


Fig. 9. Comparison of the maximum link utilizations during fast reroute phase and OFIB phase with and without optimization.

Figure 9 uses an illustration similar to Figure 6. The first bar of each network shows the best maximum link utilization when link costs are optimized for not-via fast reroute but not considering OFIB. These values are identical to the ones in the first bar of Figure 6. The second bar of each network shows the same link cost settings, now evaluated for the maximum link utilization during OFIB using the upper bound algorithm (OFIB unoptimized). These bars represent upper bounds and are thus larger or equal to the worst found OFIB value bar in Figure 6. The third bar shows the best maximum link utilization value, i.e., the lowest found upper bound, during OFIB when the OFIB upper bound is considered during the optimization (OFIB optimized).

For all networks under study the first and third bar are equal. This shows that our extended heuristic is able to avoid the maximum link utilization increase during the OFIB phase without impairing the maximum link utilization during the not-via fast reroute phase.

VII. Conclusion

In this paper, we provided an overview of the *ordered FIB update* (OFIB) loop-free convergence mechanism. It is deployed to avoid packet loss due to micro-loops during the reconvergence after a failure. Nevertheless, unanticipated temporary utilization increases that can also lead to packet loss can still occur in this phase. We demonstrated that the actual link utilization depends on the router update order, which is only partially specified by OFIB. We provided a simple and fast mechanism to calculate a tight upper bound on this utilization increase. We showed that some link loads can temporarily exceed the maximum link utilization that occurs otherwise in the network, including fast reroute and reconverged routing after failures.

These facts led us to the conclusion that the reconvergence phase should be taken into account during link cost optimization. We did that efficiently by taking advantage of the proposed upper bounds on the utilization increase. We optimized the routing for multiple network topologies to minimize the maximum link utilization in the failure-free state, the fast reroute state, and the re-converged state with and without inclusion of all OFIB stages. The inclusion of OFIB states in the optimization completely avoided the effect of temporary load increases on the maximum link utilization and did not lead to worse utilization values in the other routing stages.

Hence, we performed a link utilization analysis covering all possible routing states during failure handling and recovery and provided the first resilient IP routing optimization that considers all of these states.

Acknowledgments

The authors thank Prof. Phuoc Tran-Gia for the inspiring discussions and helpful comments.

References

[1] M. Shand, S. Bryant, and S. Previdi, "IP Fast Reroute Using Not-via Addresses," http://tools.ietf.org/html/draft-ietf-rtgwg-ipfrr-notvia-addresses-06, Oct. 2010.

[2] P. Francois, O. Bonaventura, M. Shand, S. Bryant, S. Previdi, and C. Filsfils, "Loop-Free Convergence Using oFIB," http://tools.ietf.org/html/draft-ietf-rtgwg-ordered-fib-04, Oct. 2010.

[3] M. Menth, M. Hartmann, and R. Martin, "Robust IP Link Costs for Multilayer Resilience," in *IFIP-TC6 Networking Conference (Networking)*, Atlanta, GA, USA, May 2007.

[4] M. Hartmann, D. Hock, M. Menth, and C. Schwartz, "Objective Functions for Optimization of Resilient and Non-Resilient IP Routing," in *7th International Workshop on the Design of Reliable Communication Networks (DRCN)*, Washington, D.C., USA, Oct. 2009.

[5] D. Hock, M. Hartmann, M. Menth, and C. Schwartz, "Optimizing Unique Shortest Paths for Resilient Routing and Fast Reroute in IP-Based Networks," in *IEEE Network Operations and Management Symposium (NOMS)*, Osaka, Japan, Apr. 2010.

[6] P. Francois and O. Bonaventure, "Avoiding Transient Loops during the Convergence of Link-State Routing Protocols," *IEEE/ACM Transactions on Networking*, 2007.

[7] M. Shand and S. Bryant, "RFC5715: A Framework for Loop-Free Convergence," Jan. 2010.

[8] A. Zinin, "Analysis and Minimization of Microloops in Link-state Routing Protocols," http://tools.ietf.org/html/draft-zinin-microloop-analysis-01, May 2005.

[9] P. Francois and O. Bonaventure, "Avoiding Transient Loops during IGP Convergence in IP Networks," in *IEEE Infocom*, Miami, Florida, Mar. 2005.

[10] J. Fu, P. Sjödin, and G. Karlsson, "Loop-free updates of forwarding tables," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 5, no. 1, Jul. 2008.

[11] L. Shi, J. Fu, and X. Fu, "Loop-Free Forwarding Table Updates with Minimal Link Overflow," in *IEEE International Conference on Communications (ICC)*, Jun. 2009.

[12] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP Topologies with Rocketfuel," in *ACM SIGCOMM*, Pittsburgh, PA, Aug. 2002.