Budapest University of Technology and Economics
Department of Telecommunications and Media Informatics

New Network Optimization Methods for Fast Protection in IP Networks

Levente Csikor

PhD Dissertation

Advisor:

Gábor Rétvári, Ph.D. (Senior Research Fellow)

Dept. of Telecommunications and Media Informatics and
Hungarian Academy of Science Future Internet Research Group,
Budapest University of Technology and Economics

Budapest, Hungary
2015.

...to my parents.

Abstract

Nowadays, many commercial telecom and multimedia providers have started to broadcast their contents over the Internet in order to reduce costs and reach new users. This continuous convergence is producing a more and more heterogeneous traffic with different demands as many real-time applications such as VoIP, IPTV, online gaming, etc. have become available. However, the integration has happened so fast that the network community could not adapt the IP protocol suite fast enough to keep up with the new requirements and there still exist missing components to facilitate the desired transmission quality.

One of the main concerns is that an operational network frequently suffers component failures and the currently used reactive techniques take too much time to recover the appropriate paths and bypass a failed component. Instead, faster proactive approaches can be used, whereby after a failure, the traffic could be immediately switched to an alternate route by means of precalculated backup paths.

Although there have been numerous proposals to provide faster proactive protection in IP networks, so far only the Loop-Free Alternates (LFA) technique has been widely implemented in today's commercial routers. The main idea in LFA is to ensure that, in case of the failure on the primary forwarding path, there be a suitable backup neighbor whose path to the destination is unaffected by the failure. The strength of LFA lies in its simplicity, however, this simplicity comes at the price that not all networks could be protected.

Recently, a generalization of LFA, called Remote LFA (rLFA), has been defined where, in contrast to simple LFA, not just direct neighbors but remote nodes too can be considered when searching for the backup path. Unfortunately, rLFA is equally plagued by LFA's main weakness, namely, that there are many networks that cannot be completely protected, leaving these networks vulnerable to certain failure scenarios.

The main contribution of this thesis is an analytic study of the failure case coverage attainable by LFA and rLFA in different network topologies and the development, analysis, and performance evaluation of novel network optimization techniques aimed at designing more resilient networks.

In the first part of this thesis, we investigate to what extent a network can be optimized to improve the level of protection provided by LFA with only minor changes in the topology. We examine the complexity of these problems, and we give algorithms to solve them. We demonstrate that the protection provided by LFA can be boosted close to 100% failure case coverage against single link failures, while we also show that our methods are effective against the rarer but relevant case of single node failures as well.

In the second part of this thesis, we extend the failure case coverage analysis and network optimization techniques, so far only available to LFA, to the upcoming rLFA technique, and we show that the level of protection can be improved significantly by Remote LFA compared to the protection provided by pure LFA, and we analyze which particular topological properties bound the failure case coverage of Remote LFA. Last but not least, we show network optimization techniques to provide 100% rLFA failure case coverage irrespectively of whether link or node protection is considered.

We believe that our results may contribute to the general understanding of the level of protection that the LFAs can provide, and may help hesitating network operators to reach a verdict of using any of the available approaches.

Kivonat

Napjainkban az Interneten egyre nagyobb teret hóditanak a tradicionális alkalmazások mellett a kereskedelmi távközlési és multimédia szolgáltatások, amik olyan, az eddigiektől merőben különböző és új igényeket támasztó valós idejű alkalmazások megjelenését vonták magával, mint például a VoIP, IPTV vagy akár az online játékok. Azonban ez a technológiai konvergencia olyan gyors ütemben zajlik, hogy a jelenlegi Internet eddig nem tudott teljes mértékben lépést tartani vele, és még mindig hiányoznak a kívánt átviteli minőséget garantálni képes komponensek. Ugyanis az Interneten gyakran lépnek fel meghibásodások, amiket a jelenleg is használt reaktiv technikák csak sok idő után tudnak helyreállítani. Azonban léteznek gyorsabb proaktív védelmi módszerek, melyek egy esetleges hiba után a forgalmat azonnal elkerülő útvonalakra tudják terelni.

Máig rengeteg javaslat született a probléma megoldására, de csak az ún. Loop-Free Alternates (LFA) módszer az, amely az egyszerűségének köszönhetően nem veszett el a szabványosítás útvesztőjében és elérhető napjaink útvonalválasztóiban. Sajnos az egyszerűségnek van egy hátulütője is, miszerint a módszer nem tudja garantálni a védelmet minden hálózatban. Az LFA esetén ugyanis, egy hiba észlelése után egy olyan szomszéd csomópont, amely biztosítani tudna egy elkerülő útvonalat, nem mindig létezik.

Nemrég, a védelem növelése érdekében megjelent egy általánosított módszer, az ún. Remote Loop-Free Alternates (rLFA), amely során - az LFA-val ellentétben - nemcsak a közvetlen szomszédok, hanem távolabbi csomópontok is részt vehetnek a hiba elkerülése érdekében. Ugyan a távolabbi csomópontok használata nagy mértékben javít a legtöbb helyzeten, igénybevételük lehetősége a szimpla LFA-hoz hasonlóan nem mindig garantált.

A disszertáció legfőbb célja, hogy ezen továbbra is fennálló problémák orvoslására

egy átfogó képet és teljesítmény elemzést adjon a különböző típusú LFA-kat illetően, valamint hogy megmutassa, hogy egy tetszőleges hálózatban minimális operátori módosításokkal milyen mértékben növelhető a meghibásodások elleni védelem.

A disszertáció első felében különböző hálózatoptimalizálási módszereket boncolgatunk, melyekkel jelentősen javítható az LFA által biztosított lefedettség. Továbbá tanulmányozzuk ezen megközelítések bonyolultságát és algoritmusokat javaslunk a megoldásuk érdekében. Megmutatjuk, hogy egyszeres hálózati link hibák esetében az LFA lefedettség akár 100%-osra is növelhető, valamint kitérünk a ritkább, de szintén releváns, egyszeres csomópont meghibásodások esetére is.

A disszertáció második felében betekintést nyújtunk az eltérő LFA-k közötti hasonlóságokba és különbségekbe, valamint megmutatjuk, hogy az egyszerűbb LFA-hoz képest milyen mértékben garantál nagyobb védelmet a nála általánosabb rLFA. Továbbá górcső alá vesszük, hogy mik azok a topologiai tulajdonságok amik jelentősen korlátozzák az rLFA hatékonyságát egy tetszőleges hálózatban. Végül, de nem utolsó sorban szemügyre vesszük, hogy függetlenül a meghibásodások típusától, milyen hálózatoptimalizálási módszerek alkalmazásával tehető egy hálózat teljesen védetté.

Úgy gondoljuk, hogy az eredményeink jelentősen elősegíthetik az LFA módszerek működésének megértését, valamint segíthetnek eddig hezitáló hálózati operátoroknak letenni a voksukat valamely elérhető módszer mellett.

Acknowledgements

First of all, I would like to thank my supervisor, Gábor Rétvári for all the time and energy he has put into supervising and guiding my research. His encouragement, guidance and support were indispensable to becoming a researcher in the field of telecommunications. He taught me a lot of things in the way of thinking, developing and presenting. I want to highlight that without his useful comments on all of my paper preparations, I would not have been able to write any substantial research paper. He always gave me directions and thoughts when I felt myself lost.

I am also indebted to Zalán Heszberger, who was my supervisor in my M.Sc. years and helped me make my way towards being a PhD student. During my PhD studies, he was always able to help me when I was a bit trembled.

I would like to thank to my colleagues and roommates, Zoltán Fehér, Balázs Lajtha, Attila Kőrösi, Márton Csernai, Péter Babarczi, Felicián Németh, Balázs Sonkoly, András Gulyás and all the people whose names would not fit on this paper for all their help and the excellent social events we spent together.

My work was carried out in the Department of Telecommunications and Media Informatics at Budapest University of Technology and Economics, in the Hungarian research group, called MTA-BME Lendület and in the High Speed Networks Laboratory (HSNLab), therefore I would like to thank to János Tapolcai, Róbert Szabó, Attila Vidács, Sándor Molnár for their support and advices and to Erzsébet Győri for all her help.

Last but not least, I would like to thank to my parents for supporting me through all my student years, and made it possible to me to concentrate on my studies. I also would like to thank to my brother and to all my relatives for their love and patience, without the atmosphere of such a great family, my dreams would have never come true.

Contents

Chapter 1

Introduction

Currently, the Internet has reached the level of reliability, where communication and cloud services are widely spreading among users. This gives an increasing push on service providers to operate the Internet without any interruption and slowly win the trust of most of the potential users. We expect that the reliability of IP networks will further improve in the future, and the Internet will become a critical infrastructure for the society. Reliability means that at any given time the connection is ensured throughout the network, and a failure is handled so fast that virtually no packet loss is noticed by the end-users.

Nowadays, not just Internet Service Providers (ISPs) and end-users are concerned, but many other multimedia providers started to gain a foothold in this field and broadcast digital content over IP (Internet Protocol). Moreover, traditional telephony is already being replaced by IP based telephony in order to reduce costs and provide more options, e.g., send text, media and data simultaneously during the phone call. Due to this continuous technical change and digital convergence (collectively referred to as information and communication technologies [1]), a huge number of (real-time) multimedia applications are using the IP network as a primary transmission medium, which is the first driving force for a more reliable IP communication infrastructure.

Furthermore, not just the scope of contents is growing but the number of the newly connected consumers and terminal equipments as well. The population of the world is currently growing at a rate of around 1.14% per year, and the expected population will be 8 billion in 2024[1], and, what is more, the number of connected

[1]http://www.worldometers.info/world-population/ (accessed in Nov 2013)

devices is expected to rise to nearly three times as high as the global population to 2017 [2]. Today, about two billion people are using the Internet, while six billion people are already a subscriber of some mobile services at the end of 2011 [3]. In the near future, not only human beings will be connected to the Internet all the time, but many machines used day by day will have unique IP addresses and will be accessible from anywhere. Moreover, due to the evolution of mobile infrastructure, most of the new users will access all digital content through their smartphones, increasing the traffic that has to be delivered at the same time.

Through the development of already used entertainment services, for instance, television broadcasting and Video on Demand (VoD), real-time video broadcasting will account for two-thirds of the world's mobile traffic, while the sum of all forms of video (TV, VoD, Internet, Peer-to-Peer) will be in the range of $80 - 90\%$ of global consumer traffic by 2017 [2]. Note that not just mobile phones account for mobile traffic, since in 2012 the number of mobile-connected tablets increased 2.5-fold to 36 million, and each tablet generated 2.4 times more traffic than smartphones [4].

Since the improving quality of the contents (e.g., High Definition movies, 3D, 4K, lossless audio coding) involves a growing size of media streams, the aforementioned proportion will likely grow even further. It was forecasted that the gigabyte equivalent of all movies ever made will cross global IP networks every 3 minutes, which means that it would take an individual over 5 million years to watch the amount of video that will cross global IP networks each month in 2017 [2]. Necessarily, the Internet has to keep up with these real-time applications, which require continuous and reliable connections.

Therefore, *high availability* has become an all-important factor in operational networks. However, studies over the past decade have shown that the reliability of the Internet falls short of the five nines (99.999%) availability, which is readily available in the public-switched telephone network (PSTN) today [5].

One of the key components for providing the desired reliability is supplementing routing protocols' currently used *restoration based mechanisms* with faster *protection schemes* against network failures. In this Dissertation, we study to what extent these failures can be handled in an IP network, and we investigate several approaches to improve availability if it were not adequate otherwise.

1.1 Background

In this section, we briefly recap the basic concept of routing protocols used currently in the Internet. In particular, we review the structure of the Internet, and summarize how end-to-end connection is provided between two arbitrary users.

The Internet of today is consisting of more than a billion nodes, and what is more, by means of middleboxes (e.g., NAT[2]), there are much more devices connected to the Internet nowadays. In order or handle this huge amount of equipments, the Internet is organized into a hierarchical structure, in particular, it is an interconnection of thousands of Autonomous Systems (AS), which are in turn also an interconnection of thousands of routers and links managed by a company, an organization, or an Internet Service Provider (ISP).

In order to provide end-to-end connection between any two nodes around the world, the so called Internet Protocol (IP) is used. More precisely, IP is the part of the Internet protocol suite, which is commonly known as the 4-layered TCP/IP (Transmission Control Protocol/Internet Protocol) model (see Fig. 1.1).

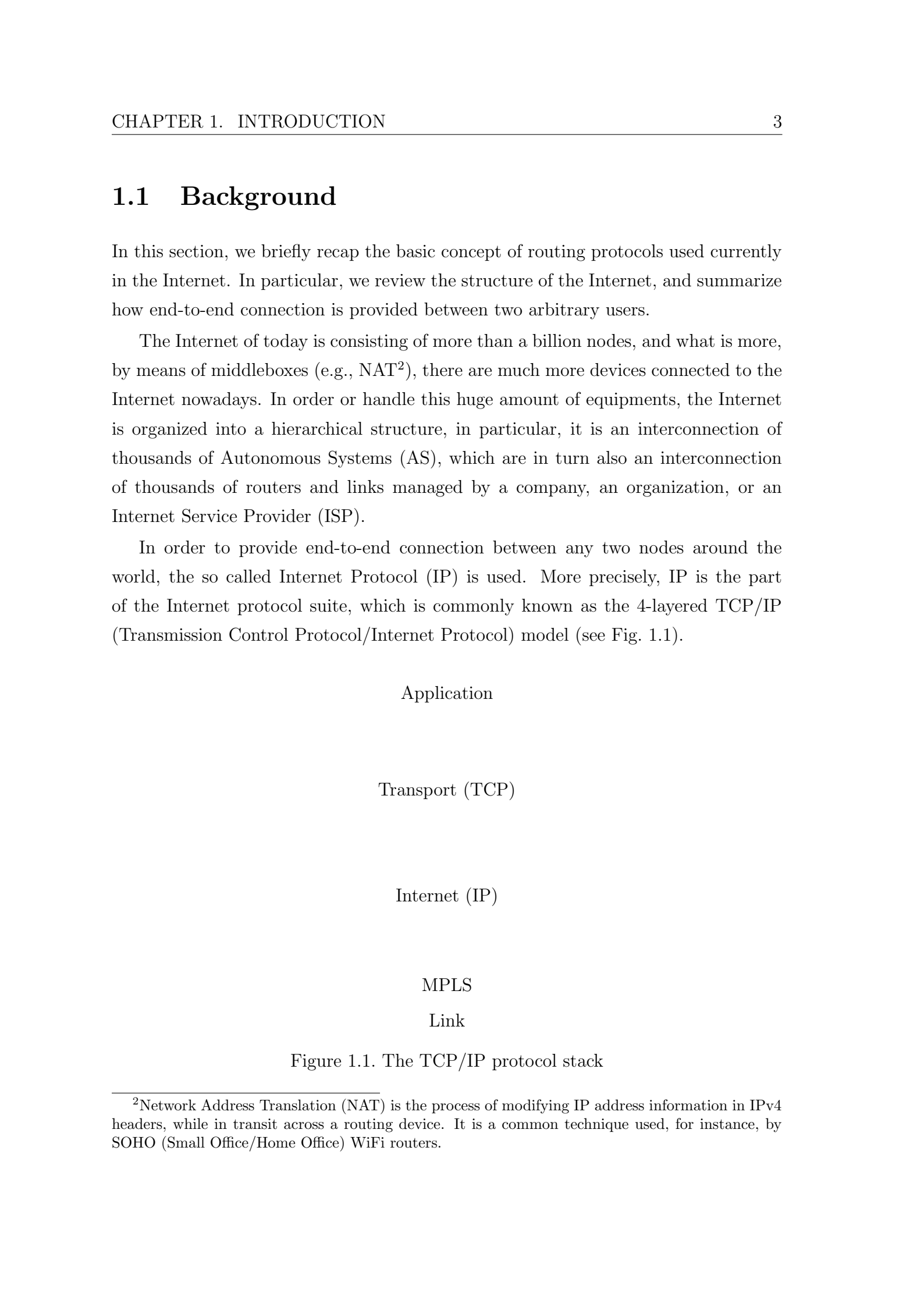


Figure 1.1. The TCP/IP protocol stack

[2]Network Address Translation (NAT) is the process of modifying IP address information in IPv4 headers, while in transit across a routing device. It is a common technique used, for instance, by SOHO (Small Office/Home Office) WiFi routers.

In a nutshell, the TCP/IP protocol stack uses encapsulations to provide abstraction of services. In general, the application layer uses a set of protocols to send data downwards the layers, being further encapsulated at each level. The Transport layer is responsible for end-to-end connections and reliability, while the IP layer is responsible for path determinations and addressing (IP addresses). However, in large enterprise networks the IP layer is often complemented with a protocol independent and scalable solution, called MultiProtocol Label Switching (MPLS, [6]), which can be considered as a 2.5^{th} layer[3], and it appears after the data link layer, but before (any) network (IP) layer. The Link layer uses a group of methods that operate on a host's link and provides the connectivity between two adjacent nodes in the network.

The data one node wants to send to another node is divided into packets, which are forwarded in a hop-by-hop manner. In order to provide a path for the communicating nodes, *routing protocols* are deployed. Then again, routing can also be divided into two main components, *intra-domain* and *inter-domain* routing. The intra-domain or Interior Gateway Routing protocols (IGPs) are responsible for forwarding packets within an AS, while inter-domain routing accounts for ensuring the connection between ASes. In an intra-domain setting, different routing protocols can be used such as OSPF (Open Shortest Path First, [8]) or IS-IS (Intermediate-System-to-Intermediate-System, [9]), however, there is only one currently used inter-domain routing protocol (Border Gateway Protocol (BGP, [10])). Intra- and inter-domain routing protocols fundamentally differ, since their objectives are different. The aim of intra-domain routing is to provide shortest paths between the nodes throughout a domain, while inter-domain routing is shaped by best commercial interests and (political) policies. Inside an AS, at most a few thousand nodes have to be treated typically, while in the larger scale of inter-domain setting hundreds of thousands of nodes are being maintained. Consequently, in an AS it is not a problem to advertise the whole topology, however, ISPs usually averse to share this information with other ASes. Thus, in order to ensure the communication among ASes, so called egress routers are deployed at the edge of a domain, which has two main reasons. First, a router inside the domain only has to know the route to the egress router in order to send data to another node situated in different domain. On the other hand, from the egress router's point of view, it only has to know the route to a remote egress router

[3]Note that the 2.5^{th} term comes from the ISO/OSI model [7].

situated in a different domain (by means of inter-domain routing), without knowing how a particular node should be reached within that AS.

In the next section, we continue our discussion with intra-domain routing and we show how path management is done.

1.1.1 Intra-domain Routing and Forwarding

An IP router has two main components, the *control plane* and the *forwarding plane*. A simplified view of an IP router is depicted in Fig. 1.2. According to the topology, the control plane defines in which direction an incoming IP packet should be forwarded by means of routing protocols[4]. In particular, it computes shortest paths towards all possible destinations[5] within the domain, termed as *prefixes*, and store this information in the Routing Information Base (RIB). The RIB, which is often called routing table, consists of full set of routes exchanged by the IGPs and it is optimized for efficient updating and other control plane methods. In contrast to RIB, the simplified extraction, called Forwarding Information Base (FIB), is situated in the forwarding plane. The FIB is optimized for fast lookup and it consists of only the necessary information to forward an IP packet. In particular, based on the shortest paths the FIB only contains interface identifiers and *next-hop*[6] information for each reachable destination network prefix [11].

Furthermore, the forwarding plane manages linecards with the practical physical interfaces, and according to their FIBs, they "put the packets on the right wires". For instance, when a packet arrives at the router, the linecard performs a lookup for the destination address obtained from the packet in its locally stored FIB, and forwards it in line with the results of this lookup. Based on the shortest path to the destination, the result is one of the neighboring nodes (a next-hop), and the interface itself whereby it can be reached. If the corresponding destination was not found in the FIB then the packets are dropped. When topology changes happen in the network, for instance when a failure occurs, the routing protocol updates the RIB, and those changes are reflected in the FIB. However, if after a failure a destination node still

[4]Besides routing, control plane is involved in other tasks as well (e.g., encapsulation, decapsulation).

[5]The calculation of the shortest paths are based on the administrative costs of the links (see details later).

[6]In IP routing, the next router along the shortest path to a destination is called *next-hop*

Figure 1.2. A simplified view of an IP router consisting of two line cards with two and one interfaces

cannot be reached from a particular node (e.g., when the network fell into two distinct parts), then packages will still be discarded.

Nowadays, *link-state routing* protocols such as OSPF and IS-IS are used within an AS. These protocols are maintaining databases (Link-State DataBase (LSDB)), which describe the entire Autonomous System.

First, each router only knows its neighbors and the distances to them based on the neighboring link metrics[7].

This information, encoded in Link-State Packets (in IS-IS) or Link-State Advertisements (in OSPF), are flooded throughout the network in order to provide a complete view of the topology for all routers. When every router has a consistent view, they calculate shortest paths to every possible destination by means of Dijkstra's Shortest Path algorithm [13]. From a graph-theoretical aspect, each router constructs a tree of shortest paths with itself as root.

Next, we show how forwarding is done after all routes have been determined by the IGP. A simple example of forwarding a packet is depicted in Fig. 1.3. Nodes marked with Rx are routers, while s, d, a, c are hosts. The colored elements should only be considered when MPLS is deployed (explained later). Here, the source node s (IP ad-

[7]Link metrics, determined by the network operators, are typically set inversely proportional to the link bandwidths (this setting is recommended by Cisco, see documentation on `ospf auto-cost` in [12]). However, in most cases link metrics are determined in such a way that fulfills a certain goal, which is a network optimization task.

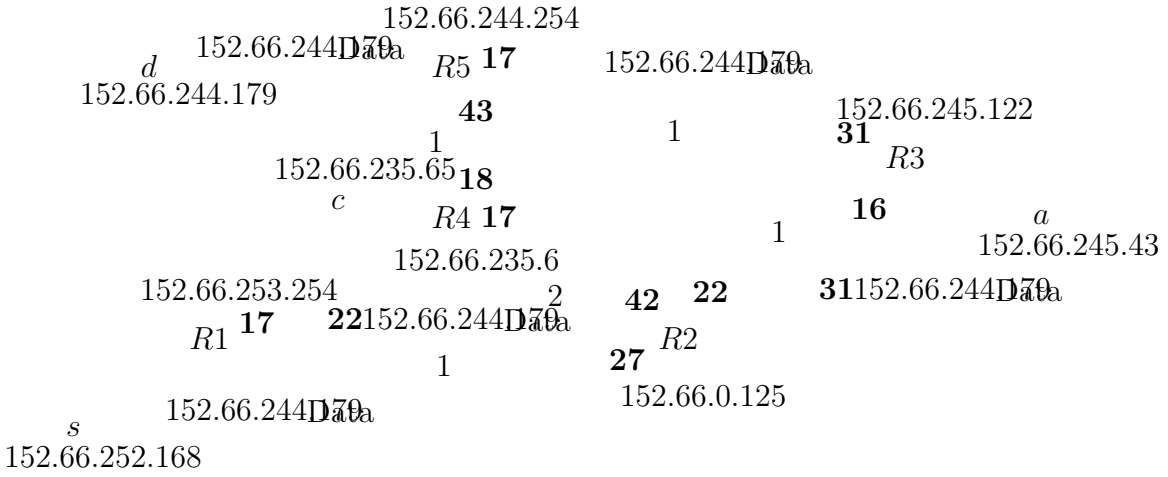


Figure 1.3. Basic example of an IP packet flow from s to d. The shortest paths from node s are marked by solid black arrows. Routers are denoted by Rx. The black labels are the administrative link costs, while the colored labels close to the routers are MPLS labels

dress: 152.66.252.168) wants to send a packet to node d (IP address: 152.66.244.179). According to the shortest path from s to d (denoted by the solid thick arrows), the concerned routers examine the packet looking for the destination's IP address in order to pass the packet towards it. After $R1$ examines the packet header, according to its FIB, it passes the packet towards its next-hop $R2$. Similarly, $R2$ also examines the packet header in order to determine that the next-hop is $R3$. Then, $R3$ will pass the packet to $R5$, from which the packet will reach its destination d. Note again that the shortest paths are determined by routing protocols.

However, there are networks where, in order to provide a protocol independent and scalable solution, MPLS is deployed. The original aim of MPLS was to reduce the number of IP lookups required for finding a particular destination. MPLS solely uses labels for path determination, without the need to examine the packet itself[8]. This allows one node to create an end-to-end circuit, called Label Switched Path (LSP), across any type of transport medium, using any kind of protocol.

Briefly, each router (termed as Label Switched Router (LSR)) assigns a locally

[8]Historically, IP lookups required many memory accesses just to route a single packet.

significant and locally unique[9] label to every destination prefix, and these labels are distributed to the neighbors saying that "if you want to reach a certain destination d use my label l". Besides Resource Reservation Protocol with Traffic Engineering (RSVP-TE, [14]), Label Distribution Protocol (LDP, [15]) is the most commonly used for distributing such labels. Similarly to the operation of pure IP routers, LSRs have a LIB (Label Information Base) in the control plane, and LFIB (Label Forwarding Information Base) in the forwarding plane. If an incoming packet is a simple IP packet, then it is forwarded according to the FIB and a label may be pushed onto the packet. An incoming labeled packet, however, is forwarded using the LFIB.

In the network depicted above (Fig. 1.3), each router Rx has assigned labels to each destination prefix according to the shortest paths obtained from the IGP. For sake of easier comprehension, not all labels are shown. By means of LDP sessions, $R2$ informs $R1$ to use label 22 to reach destination prefix 152.66.245.0/24[10], while packets destined to prefix 152.66.235.0/24 should use label 42. Similarly, $R3$ "tells" $R2$ to use label 31 towards d. Other routers proceed likewise.

During packet traversal, the label stack (denoted by colored headers in the packet) varies hop-by-hop according to the desired labels. For instance, if $R2$ receives a packet with label 22, it swaps the label to 31, and forwards it to its next-hop $R3$. Besides label swapping, MPLS-enabled routers do label push and pop operations as well indicating the head and the tail of an LSP.

When a router is the tail of an LSP, e.g., the case of $R5$ and destination d, then from $R5$'s point of view several possible operations can be done. In particular,

- $R5$ assigns a label, say 19, to node d. When a packet is received with label 19, it will pop the label and performs another lookup in its FIB. According to the results of this second lookup, $R5$ will pass the packet to d. Note that if $R5$ receives a packet with unknown label, then it proceeds similarly.

- One may have noticed that in the previous case $R5$ needed to perform two lookups, which is not an optimal way of forwarding labeled packets. Therefore, in order to get rid of this double lookup, *penultimate hop popping* is used. It means that, $R5$ assigns the reserved[11] label 3 ("implicit NULL label") to

[9] Other routers may use the same label but for different purposes.
[10] 152.66.245.0/24 denotes all the IP addresses in the range from 152.66.245.1 to 152.66.245.255.
[11] Note that there are several reserved labels for indicating special purposes (see details in [16]).

destination d, which indicates a label pop operation in $R3$. In this case, $R3$ will remove the label from the packet and sends it towards $R5$, where, consequently, only an IP lookup needs to be performed.

Note that $R1$, $R3$ and $R4$ also assigns label 3 to all destination prefixes to which they are directly connected. For a comprehensive overview, see [17].

If for some reason bypassing the default shortest path is required, so called tunneling mechanisms can be used such as IP-in-IP tunneling [18]. In our example, if $R1$ wishes to send packets to $R5$ through $R4$, then the original packet is being encapsulated into another IP packet, wherein the destination IP is $R4$'s IP address[12]. After $R4$ decapsulates the packet, it recognizes that the remaining part of the packet is still an IP packet destined to node d. Therefore, it will send the packet to $R5$, which is the next-hop of $R4$ towards d.

This tunneling, however, can be done by relying solely on MPLS labels. By means of Targeted LDP sessions, LSRs can obtain labels from remote nodes as well, in order to learn which particular label a nonadjacent LSR uses to reach a certain destination.

To illustrate this case, consider again the example above. First, $R1$ initiates a Targeted LDP session to $R4$, from which it will become aware of that $R4$ uses label 18 to reach $R5$. Thus, $R1$ pushes this label onto the packet, then $R1$ also pushes $R2$'s label (42) into the label stack in order to reach $R4$ itself. Accordingly, after $R2$ receives a packet destined to node $R4$, it will pop the label due to penultimate hop popping and send the packet towards $R4$. Then, $R4$ will recognize the remaining label 18, which will be popped again and the packet will be sent to $R5$, from where it finally reaches d.

Bear in mind that MPLS is an extension to the IP protocol suite, and it is not required for standard operations and routing. There are many pure IP networks deployed nowadays, especially in smaller ASes.

Nonetheless, once a connection is set up between two arbitrary nodes, it is not guaranteed that the path, along they send packets, will not change. Unfortunately, an operational network often suffers component failures that occur frequently due to various reasons such as physical interruptions, flapping interfaces[13], etc. Regardless of whether a link or a node fails, routing protocols are responsible to react to the

[12]Note that in this case the size of the IP packet increases.

[13]A failure of an interface can cause the router to announce it alternately as “up” and “down”.

failures and find an alternate route in order to recover connections among affected nodes.

1.2 Restoration Mechanism of IGPs

In an intra-domain setting, the failures are handled by the IGPs, which adopt a restoration-based resilience approach. Thanks to the different network layers, there are numerous mechanisms are available for adjacent routers to detect a failure. For instance, in the physical layer the absence of the light in the optical fiber indicates that a failure occurred, or the transport layer can also infer failure by means of a certain ratio of packet losses. In case of IGPs, adjacent routers send periodic HELLO messages to each other to indicate that they are up and running. If a router misses a fixed number of HELLO packets from a neighboring node, it declares the adjacency to that router down. However, this detection time is typically greater than one second, and it often lasts longer [19]. In order to quickly react to failures, more rapid detection is necessary.

Therefore, *Katz et al.* proposed Bidirectional Forwarding Detection (BFD, [20][14]), in order to provide low-overhead, short-duration detection of failures in the path between adjacent forwarding engines. One of the main advantages of the solution is that it can be implemented on the linecards themselves providing much faster failure detection without impact on the CPU, i.e., only the forwarding plane is involved. On the other hand, with BFD not only link down and up events can be detected, but observing Layer 3 failures, e.g., router crash, is also feasible.

After detecting a failure, the information about the failed component is distributed throughout the domain so that every router can recalculate the shortest paths with the failed component removed. The whole process is called *re-convergence*, and, depending on the network size (number of hops between any two routers), congestion, computational capabilities of the routers, and several other factors it can take between 150 ms and a couple of seconds [21, 22]. Note that MPLS convergence (assigning and distributing new labels) occurs immediately after the routing protocol has converged.

A comprehensive study on re-convergence was presented in [23], where all the factors that affect the convergence time were deeply studied. The authors characterized

[14]The Internet Draft became a standard in 2010 (RFC 5880)

the convergence time as $D + O + F + SPT + FR + DD$, where the detection time (D), the LSP origination time (O) and the distribution delay (DD) are smaller thanks to BFD. The flooding time (F) highly depends on the network topology, while the shortest path computation time (SPT) depends on the number of nodes and their calculating efficiencies, however, this time can be sped up by using incremental SPT computations instead. Finally, the time of updating the RIB and the FIB (FR) is the most significant factor as it depends linearly on the number of prefixes affected by the topology change. It was found that FR time can be improved by incremental FIB updates, while faster SPT can be achieved by advertising fewer prefixes in the IGP. However, even if faster convergence time can be reached, it is not guaranteed that these optimized methods are available in an arbitrary network, and the size of the network still has a significant impact on sub-second convergence.

From the aspect of reliability, the re-convergence process has other consequences that have to be taken into account. In particular, it has *two subsequent effects*: after a failure, (i) packets are dropped due to the absence of working paths[15], and (ii) temporary micro-loops can occur until all routers converge on the same view of the network. These two phases are explained in Fig. 1.4. Suppose that packets from node s to node d traverse node e (1). If the link between node e and d fails (2), then, after

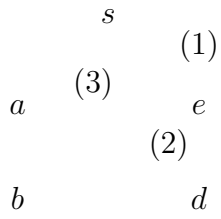


Figure 1.4. A sample network for explaining re-convergence process

noticing the failure, node e distributes this information and starts to recalculate its shortest paths. During this recalculation phase packets are dropped at node e, since it cannot reach d. After node e has a complete view of the new topology, it's new shortest path to node d would traverse node s, a and b before reaching the destination (3). However, if node s did not have the same view yet, then it still assumes that node d can be reached via node e, and it will send the packets back to e causing a loop.

[15]Note that if multiple equal cost paths were used at the same time (Equal Cost Multiple Paths (ECMP)), packets can be detoured on the remaining working path(s).

Nevertheless, we can assert that these link-state routing protocols are quite robust and efficiently act upon topology changes. Nowadays, the increasingly used real-time applications, however, cannot tolerate arbitrary re-convergence times, and the connection between a pair of nodes may become intermittent due to a failure or a manual maintenance. In order to provide a certain quality of service (QoS) for such failure sensitive applications, more and more stringent Service Level Agreements (SLAs)[16] are contracted between ISPs and customers. The case of failing to fulfill these requirements by an ISP has a serious consequence usually expressed in terms of a fee. Clearly, these restrictions make an additional push on ISPs for operating their networks more reliable by means of using proactive and faster methods, instead of the slow reactive mechanisms provided by the standard link-state routing protocols.

Before digging into fast protection techniques, we need to understand the characteristics of the failures as which element or phenomenon causes the failure itself, which part of the network is the most critical or even how long do failures last. Therefore, in the next section we give an introduction to IGP measurement studies characterizing different types of network failures.

1.3 Failure Characterization

In the absence of empirical data, initially failure characterizations were based on simulated and/or theoretical networks [26, 27] and arbitrary synthetic failure models [28, 29, 30]. As a first approach to resolve this issue, a direct method of tracing down whether a link is down or up was to attempt to use it. In commercial networks, therefore, periodic end-to-end probes [31] were conducted to localize failures.

Later, as a more fine grained measurement mechanism, operational logs and SNMP (Simple Network Management Protocol)[17] queries were combined in order to locate and analyze failures [32]. It was observed that in an inter-domain setting, contrary to the five-nines availability (99.999%) of the PSTN services, an order of magnitude lower availability is provided between three ISPs. In particular, by the end of thirty days of the study, approximately 75% of all the routes from all three providers had

[16]SLA specifies, usually in measurable terms, what services are provided from an ISP. Such metrics are percentage of availability, latency, number of simultaneously servable users, etc.

[17]SNMP is a component of the well known TCP/IP suite defined by the IETF (Internet Engineering Task Force). It consists of a set of standards for network management.

failed at least once. On the other hand, in the intra-domain setting, 40% of all interfaces experienced some failure within an average of forty days. During the one-year case study, it was shown that the largest category of failures at around 16% refers to scheduled or unscheduled emergency upgrade of software or hardware; or router configuration changes. They also observed intermittent failures that mysteriously resolved themselves before an engineer could investigate the cause. The main observation of the study is that most routing problems stem from *human error and misconfiguration* of equipments.

In order to find potential errors before deployment, in [33] a tool called *rcc* was implemented to parse BGP router configs. By means of this tool, more than 1000 BGP configuration faults were detected in real-world, deployed configurations from 17 different ASes[18].

In [35], the behavior of OSPF was described in a large enterprise network. The authors tracked more than 200 routers over a month. Although the aim was to study OSPF behavior itself, this study also provides a valuable insight into the characteristics of failures. In particular, it was observed that the majority of link failures in the network were caused by a *single misconfigured router*.

A similar study on monitoring OSPF behavior in a regional service provider network was conducted by Watson *et al.* [36]. A network consisting of fifty routers, including internal and customer links, was examined over one year. It was observed that *only a small portion of routers* contribute disproportionately to network instability, and *flapping links* are the predominant source of instability.

In 2004, Marcopoulou *et al.* [37] studied failures in the Sprint backbone[19]. They collected six months of IS-IS routing protocol messages and monitored hundreds of interconnected nodes. In addition to providing characterization of time-to-failure and time-to-repair distributions for the Sprint backbone, they observed that only 2.5% *of all links are responsible* for more than half of the individual failures. Furthermore, many of these links exhibited *flapping behavior*. Overall, their results showed that 80% of all link failures are unplanned and 30% percent of those involved shared link risk groups. They extended their work with analysis of a shorter time period (one

[18]Note that parsing config and log files is a useful technique not just for network failure characterization, but for detecting any misbehavior of any service. In [34], syslog records were analyzed to identify anomalies in datacenter operations.

[19]https://www.sprint.net/ (accessed: June 2014)

month) in order to understand to what extent the statistical characteristics vary with time [38]. In this shorter period, they observed that 20% of failures happened during a period of scheduled maintenance, while 16% and 11% of unplanned failures are shared among multiple links and can be attributed to router-related and optical-related problems, respectively.

Similarly to [37, 38], it was shown that a huge amount of unplanned failures affects *only links* and almost the half of these failures are *transient*, i.e., they last less than a minute [39].

Another measurement was conducted in a campus network consisting of 40 routers and 373 switches using SNMP-based event notifications and syslog event logging [40]. During the nine month analysis period, 1.8 million link failure events were observed, most of them again caused by *flapping links*.

In order to better understand the underlying causes of intra-domain routing instability, a joint analysis was made from 2005 to 2007 between a VPN (Virtual Private Network[20]) provider and the Internet2 (the US research backbone) network [41]. The analysis revealed that the largest fraction of routing changes in Internet2 is caused by scheduled maintenance, while the majority of the VPN provider instabilities are unplanned. Furthermore, these unplanned events lasted longer than scheduled "failures", and most of them were caused by hardware and circuit problems, and only one-third of failures occurred by software bugs.

Last but not least, Turner *et al.* [42] used a combination of structured data (router configurations and syslog files) and e-mail logs to analyze over five years of failure events in a large regional network (CENIC - Corporation for Education Network Initiatives in California) consisting of over 200 routers. They also found that hardware-failures are more prevalent than software bugs, and what is more, more than 50% of link failures were caused by *only five links*.

From the studies conducted so far, we can conclude that *most of the failures* in an operational IP network are caused by only a handful of *flapping links*, and these failures often last longer than other topology changes caused by pre-planned maintenance. Bear in mind that flapping interfaces force frequent recalculation of the topology and prevent the network from converging this way causing constant high

[20]A Virtual Private Network extends a private network throughout the Internet enabling a computer to send data to others directly as if they were in the same subnet.

overhead for IGPs (increased signaling traffic, CPU load, etc.). Furthermore, these results also suggest that there is only a small portion of "hot-spots" that should require special attention, making it possible to use simpler and therefore effortlessly deployable protection mechanisms (discussed later in Section 1.4) that may otherwise would not guarantee 100% availability. It should be noted that from the real-time applications' point of view, it was already observed in 2002 that most service disruptions of real-time VoIP applications do not occur due to insufficient resources or application problems, rather because of routing changes [43].

1.4 IP Fast ReRoute Techniques

From the previous discussion, we can conclude that with current router technologies it is definitely possible to achieve sub-second IGP convergence ($\sim 200 - 300ms$) without any compromise on stability. Unfortunately, this delay is still too much for applications with real-time demands ($50ms$). Therefore, in order to provide faster recovery, standard IGPs have to be complemented with other techniques.

In 2004, the IETF defined the *IP Fast ReRoute Framework*[21] (IPFRR, [44]). IPFRR is based on two major principles: *local rerouting* and *precomputed detours*. Local rerouting means that instead of notifying every other router about the failure, the adjacent router tries to (re)solve the problem locally, i.e., reroute the packets to another node this way bypassing the failed component. Precomputed means that the mechanism is proactive and the alternate backup paths are installed long before any failure occurs. Thus, the IPFRR techniques convert the restoration scheme, standard in IP networks today to handle outages, into a faster proactive protection mechanism [45].

In this section, we briefly review the important proposals appeared so far, and besides their advantages, we also show their drawbacks and discuss whether they can be deployed in today's Internet.

The first technique proposed, implemented and already deployed is MPLS label swapping forwarding (*MPLS FRR*, [46]). In cases when an IP network does not use MPLS to actually forward packets, it is possible to use MPLS with RSVP-TE for protection only. However, the main disadvantage is that many network operators rely

[21] The IP Fast ReRoute Framework became a standard in 2010 (RFC 5714).

on MPLS/LDP (Label Distribution Protocol) exclusively, therefore they can protect only a small portion of operational networks.

The first pure IP-based protection technique, called *Loop-Free Alternates (LFA)*, was proposed in the IPFRR framework [44], which was more precisely defined later in a separate document [47] and became a standard in 2008. In LFA, when the connectivity to a next-hop is lost, all the traffic is rerouted to an alternate next-hop, called a *loop-free alternate* that still has a path to the destination, which is unaffected by the failure. These alternate next-hops are selected in a way as to guarantee that packets will not be passed back, since that would lead to a micro-loop. However, such alternate next-hops do not always exist, since the availability of an LFA greatly depends on the actual topology and link costs. Thus, in most network topologies, not all next-hops can be protected with LFA, leaving the network vulnerable to certain failure scenarios.

As an extension to Loop-Free Alternates, a method called *U-turn Alternates* was defined in [48], wherein if a router adjacent to a failure has no loop-free alternate, it still sends the traffic to one of its neighbors and informs that neighbor to use its loop-free alternate to reach the destination. However, this is the main drawback of the method, since, for instance, using extra bits in the IP header is essentially ruled out by most network operators as it would require modifications to router interface cards. Furthermore, since the efficiency of this technique also depends on the underlying topology, it still cannot protect every single network failure.

In order to expand the set of loop-free alternates provided by the last two mechanisms mentioned above, *protection tunneling* was proposed [49]. A tunneled repair path tunnels the affected traffic to some staging point in the network, from which the packets will traverse to the destination using normal forwarding without causing loop. To reach this end, several tunneling schemes can be used in IP networks, for instance, L2TP [50], GRE [51], IP-in-IP [18].

A closer look to the cases when a suitable tunnel endpoint cannot be reached often shows that only one extra hop would be needed. The method called *directed forwarding* aims to overcome this by permitting the tunnel endpoint to directly pass the packets to its neighbor regardless of the shortest paths. The main advantage of the protection tunnels with directed forwarding compared to the basic protection tunneling mechanism is that in case of symmetric link costs, each link throughout the

network can be protected [52].

Due to the requirements needed for deploying such tunneling mechanisms, which are not always available, a plethora of other proposals has appeared to overcome or alleviate somehow this drawback, or to approach the problem from a completely different point of view.

In the case of *O2 routing* [53], each router has alternate paths through at least two distinct next-hops to each destination, in order to facilitate local failure reaction and loop-free forwarding. However, the calculation of the paths are not based on shortest paths.

The concept of detecting the packet route via extra information of the incoming interfaces, used by U-turn Alternates, is generalized in *Failure Insensitive Routing* (FIR, [54, 55]). It was improved several times to handle multiple and different kind of failures (Failure Inferencing based Fast Rerouting (FIFR, [56, 57]), Blacklist-Based Interface-Specific Forwarding (BISF, [58]), Loop-free Failure Insensitive Routing (LFIR, [59, 60, 61])). The main idea is that if a node receives a packet through an unusual interface, it can infer implicitly that, due to a failure, the packet has not traveled along its default shortest path.

In *Not-via*[22], when a failure occurs packets are forwarded on an explicitly defined detour, which definitely bypasses the failed component, i.e., if an arbitrary node s wants to send a packet to node d, and the link to the next-hop e or e itself fails, then s passes the packet towards d *not-via* e [62]. Thus, this mechanism requires additional (not-via) addresses for which there is no standardized protocol. Similarly to FIR, Not-via has also been improved several times (rNotVia, [63], *lightweight Not-via*, [64]) in order to reduce the computational costs and forwarding table entries.

A different approach for providing complete failure case coverage against both link and node failures is called *Multiple Routing Configurations* (MRC, [65][23]), wherein a small set of backup network configurations is used. Thus, in case of a failure, an adjacent router detects it and marks the packets with a backup configuration identifier designating an overlay topology that does not contain the failed component.

As an improvement, *relaxed MRC, [67]* was proposed, which improves resource

[22]The initial version of Not-via was improved many times before it became a standard (RFC 6981) in 2013, but now it is considered as a purely informative document, which does not constitute a protocol specification.

[23]An extended version of the proposed technique can be found in [66].

utilization and provides fast reroute in the presence of multiple correlated failures as well.

A similar approach is [68], wherein the protection and restoration is provided by *distributed multipath routing (DMR)*. The main idea is that if multiple paths exist in the network due to load balancing, then they can be used as backup routes as well.

Failure-carrying Packets (FCP, [69]) is based on a well-defined set of potential links that does not change very often. This set is called Network Map, and since all routers have a consistent view of it, all that is required to be carried by the packets is the information about which of these links have failed.

The methods Loop-Free Alternates Paths (LFAP, [70]) and Fast Path Notification (FPN, [71]) use explicit signaling to notify routers about the failures avoiding the need of modifications to standard IP forwarding.

Last but not least, a centralized routing solution, called *Protection routing* scheme, was proposed in [72], where a central server precomputes forwarding decisions for common failure scenarios and download this information into the routers. Thus, if a failure occurs, the appropriate forwarding state is already available locally. This centralizing eliminates uncertainty and many inconsistencies, and offers flexibility in computing routes that meet different criteria. However, using a central node has its own disadvantages, since besides that it increases the latency and introduces a completely new routing architecture, the designated node may also fail leaving the whole network unprotected[24].

It must be noted that networks are usually multilayered, thus the physical failure of a link (or node) may cause failures in a set of virtual links (or nodes) in the overlay topologies[25], which also has to be covered. Therefore, in order to provide resiliency in such cases, [73] uses SRLG-disjoint path pairs in optical networks to avoid the failures. Besides, many other approaches were proposed to provide protection in the optical layer. Interested readers are referred to [74], but note that a comprehensive overview for failure recovery in the optical layer is beyond the scope of this Dissertation.

Due to the complexity of the aforementioned techniques, it is no wonder that so far only LFA has made its way into commercial IP routers [75, 76, 77]. However, LFA

[24]Note that a similar approach has been nowadays adopted in the emerging field of Software Defined Networks (SDN), where the control and forwarding planes are completely decoupled.

[25]This phenomenon is termed as Shared Risk Group (SRG), and if only link failures are considered then it is called Shared Rish Link Group (SRLG).

cannot protect each next-hop in all networks. As a workaround, the IETF suggested to use LFA and Not-via side-by-side in the cases when the former does not deliver sufficient levels of protection [47, 62]. However, the authors of [78] proved that in real networks, where the sheer size of the IP forwarding tables and traffic engineering also play important roles, this combined method does not provide any significant advantages over pure Not-via.

In order to increase the LFA coverage, the authors of [79] proposed the method Enhanced-LFAs (eLFA), but it requires protocol changes and it is more complex than normal LFAs, defeating their major advantage over other IPFRR solutions.

Recently, the IETF has published a generalization of LFA, called the Remote LFA [80] (rLFA) in order to improve the failure coverage provided by simple LFA. Since it is based on LFA, it is already available in today's routers [81]. The main idea is that, in case of a failure, not only direct neighbors can be used as a potential loop-free alternate but further remote nodes as well. These Remote LFA staging points are reached trough IP tunnels, but these tunnels are restricted to shortest paths as well. Note that in an MPLS/LDP (MultiProtocol Label Switching–Label Distribution Protocol) enabled network, these tunnels are freely accessible via a simple label stack (more details are discussed in Chapter 5). Yet, even if Remote LFA can produce higher failure case coverage than pure LFA, the level of protection still depends heavily on the underlying topology and link costs.

A summarizing overview of the IPFRR techniques mentioned above can be found in Table 1.1. One can easily observe that most of the approaches, aimed to provide 100% protection, introduce some forms of in-band signaling, for instance, by means of using extra bits in the IP header. This brings additional complexity into routing if the extended IP header does not fit into the MTU (Maximum Transfer Unit[26]) causing packet fragmentation and time-consuming reassembly at tunnel endpoints. On the other hand, these extra bits may be completely ignored by the routers.

Furthermore, these methods often change IP's pure destination based forwarding paradigm, since besides the destination's IP address, other information has to be also considered when deciding in which direction a packet must be forwarded. Hence, these techniques remained proposals only.

[26]In computer networking, the maximum transmission unit (MTU) is the size of the largest protocol data unit that the layer can pass onwards.

	IPFRR techniques		
Status	Proposal	Internet Draft	Standard (RFC)
Deployed		rLFA	LFA
Changes IP's destination based forwarding and/or introduces some forms of in-band signaling	O2, FIR, FIFR, BISF, LFIR, MRC, rMRC, directed multipath routing, FCP, PR, eLFA, directed forwarding	U-turn alternates	
Explicit out-of-band signaling		LFAP, FPN	
Tunneling	rNotVia, lightweight Not-Via	protection tunneling	Not-Via

Table 1.1. Summary of the important IPFRR techniques

In contrast, the concept of using explicit out-of-band signaling does not require completely new protocols, however a separate signaling mechanism dedicated to IPFRR is needed to make this approach work.

In case of tunneling, we have seen proposals, Internet Drafts, and standardized solutions as well. The main advantage of these methods that by means of tunneling not just direct neighbors can be considered in order to bypass a failed component.

Accordingly, it is obvious why only LFA and rLFA have become available in commercial routers, however, initial deployments confirmed that in many operational networks LFA indeed does not guarantee protection for all failure scenarios [82]. The case of rLFA is more difficult, since as far as we know, there is no information available about how it performs in different network topologies. We believe that instead of developing a new IPFRR method, it rather be worth to focus and rely on the existing techniques and optimize the underlying network topology in order to reach the desired reliability.

1.5 Level of Protection Provided by LFAs

In this section, we briefly review the failure case coverage that LFA can provide in a simple network, and we investigate to what extent the usage of rLFA can improve this further.

In LFA, when a failure occurs, the adjacent router tries to pass the packet to an alternate neighbor that still has a functioning path to the destination. It means that this neighbor will not pass the packet back, this way avoiding the formation of loops. From this property comes the name Loop-Free Alternates.

Consider the network with unit link costs depicted in Fig. 1.5, where the solid lines mark the IP network topology, whilst thicker solid arrows indicate the shortest paths from s to d, d' and d'', respectively.

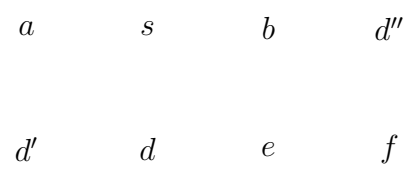


Figure 1.5. Simple network topology with unit link costs

Suppose that node s wishes to send a packet to node d. If suddenly the link (s, a) fails, s needs to find an alternate neighbor, who won't pass the packet back. Fortunately, node b fulfills this requirement, since b's shortest path to d goes through node e ($b \to e \to d$). In this case, b is a link-protecting LFA[27] for source-destination pair (s, d).

Next, consider node d' as a destination and suppose that link (s, a) fails again. In this case, node s cannot send the packets to node b, as b has ECMP (Equal Cost Multiple Path)[28] to destination d' and, as it does not know about the failure, it can send the packet back to s causing a loop.

It should be noted that in case multiple shortest paths two main scenarios have to be taken into account:

[27]There are different LFA protection schemes according to the type of the failed element, e.g. node-protecting LFA (see details later)

[28]Node b has two shortest paths (with costs of 3) to node d', namely $b \to s \to a \to d'$, and $b \to e \to d \to d'$.

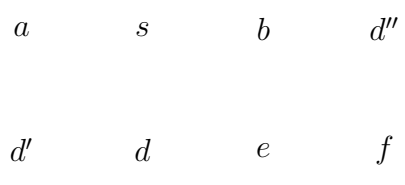
- From the aspect of the repairing node, when it has multiple shortest paths to a given destination (this is the case when node s wants to send a packet to node d in the network depicted in Fig. 1.5, but contrary to our aforementioned scenario there is no explicitly chosen single path ($s \rightarrow a \rightarrow d' \rightarrow d$) for forwarding), and one or more of the alternate paths do not traverse the failed component, they could trivially be used as repair paths (see [44] for more details). In our example, node s only needs to detour the packet to the other working path ($s \rightarrow b \rightarrow e \rightarrow d$). In this case, there is no need for the method LFA, however, such alternate next-hop fulfills LFA's strict criteria (see details later in Section 4.1).

- The other case is when after a failure the examined neighbor has more shortest paths to the destination, as the case of node b in the aforementioned example, when the traffic flow from node s to node d needs to avoid the failed link (s, a), but one of the shortest paths of node b to node d would traverse the failed component. From the LFA specification's point of view, in this case node b does not fulfill the strict criteria (again, see details later in Section 4.1), since irrespectively of whether ECMP is enabled or not at that router, if multiple paths exist, the repairing node (in our example, this is node s) could not predict which path or paths will be used by the alternate neighbor (node b) for forwarding.

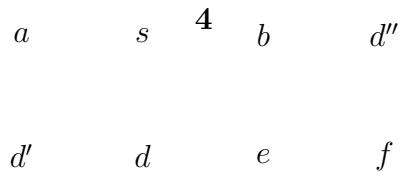
So, in the latter case the given source-destination pair cannot be protected via standard LFA. However, if a tunnel is created between s and e (marked by black dashed line in Fig. 1.5), then e, now being a direct neighbor of s, would become an LFA for d, thereby protecting the link (s, a).

Consequently, when a link cannot be entirely protected with local LFA neighbors, the protecting router seeks the help of a remote LFA staging point. Note that this tunnel is restricted to shortest paths and only used as a detour, so it does not affect the normal flow of traffic in any ways.

Next, consider node d'' as a destination and assume that the cost of link (d', d) is 2. Note that in this case according to the shortest paths, node s could only reach node a and d' without traversing link (s, b). The shortest path from s to d'' goes through node b. Under the assumption that link (s, b) fails, it cannot be protected for a lack of a suitable tunnel since all nodes (d, e, f), whose shortest path surely does



(a) LFA Graph Extension



(b) LFA Cost Optimization

Figure 1.6. Examples of different network optimization techniques to improve the failure case coverage

not go through (s, b), can only be reached from s through (s, b) itself.

This suggests that while the use of rLFA definitely can provide higher protection level than pure LFA, it still does not facilitate full protection for all failure cases in a general topology.

1.6 Network Optimization

As observed, there are certain failure cases that can be protected neither by LFA nor by Remote LFA. This calls for developing network optimization tools to tune the network topology in a way as to increase the number of failure cases protectable by LFA. There are various approaches to reach this end. One way is *LFA network design*, which aims to design LFA-friendly network topologies right from the outset [83]. Another approach is *LFA Graph Extension*, where the task is to augment the network topology with a few number of new links to boost LFA coverage [84]. Consider the network depicted in Fig. 1.6(a). As it was mentioned above, suppose again that node s wishes to send a packet to node d', and suddenly the link (s, a) fails. We have seen that in this case s has no loop-free alternate to quickly reroute the packet on a detour. However, if the network were augmented with link (s, d), then the link (s, a) would be protected by node d. Note that the source-destination pair (s, d'') would also be protected by link (s, d), which was unprotected even if Remote LFA was enabled.

On the other hand, *LFA Cost Optimization* asks to construct IGP link costs in a way as to maximize the number of possible failure cases protectable by LFA [85, 86, 87, 88]. Consider Fig. 1.6(b) and assume the same scenario mentioned above. In this case, if the cost of the link (s, b) were 4 instead of 1, then node b would not pass the packet back to s, this way becoming an LFA for (s, d'). Note again that with this slight modification the source-destination pair (s, d'') would also become protected.

Furthermore, *Combined LFA network optimization* asks for both adding new links and optimizing link costs to the same end. For instance, after the link (s, d) is added to the network in order to protect the failure of link (s, a), then the link (s, d) became unprotected, since passing the packet towards node a or node b would not bypass the failed link. However, if we change the cost of the link (s, b) to 4 as well, then b will protect the failures of both link (s, a) and link (s, d).

While improving IP resilience is a recurring theme in the literature (see [22] for deflection routing, [89] for O2, or [72] for a review), for the specific case of LFA only the joint optimization of network performance and resilience has been investigated previously [85, 86]. Thus, at the moment very little understanding is available as to how much *LFA-based IP Fast ReRoute* is suitable to protect an IP network and to what extent this can be improved.

Therefore, one of our main goals is to study both LFA Cost Optimization problem and Combined LFA network optimization problem, show their complexity and give efficient algorithms to solve them in order to provide high LFA failure case coverage in an arbitrary network.

On the other hand, this Dissertation is also aimed at investigating to what extent the failure case coverages can be improved by using rLFA instead, what are the fundamental lower and upper bounds on failure case coverage, or even how this can be improved.

1.7 Structure of the Dissertation

In the previous sections, we briefly recalled the basis and fundamentals of IP routing and discussed how they react to topology changes. We showed that single link failures occur frequently, and discussed important faster protection schemes proposed

so far with special attention on their advantages and disadvantages. After presenting that the deployable LFA and rLFA cannot protect every single failure case, we shortly demonstrated to what extent the coverage of LFA or rLFA can be improved by optimizing the underlying network topology.

The rest of the Dissertation is structured as follows. In Chapter 2, we summarize the main research objectives of this Dissertation, we demonstrate our methodology, and formulate our general assumptions.

After we briefly overview related work in Chapter 3, in Chapter 4 we explain LFA in detail with illustrative examples and show that the failure case coverage can be significantly improved by optimizing link costs. Besides, we also show the complexity of this problem, and give exact and approximate algorithms as well to solve it. Furthermore, we demonstrate that unifying *LFA Graph Extension* and *LFA Cost Optimization* is an efficient way to give an adequate intermediate solution for network operators, who cannot afford to use one of them exclusively.

In Chapter 5, we deeply analyze the failure case coverage attainable by Remote LFA. We show its fundamentals, and we point out that the protection provided by rLFA is significantly higher compared to pure LFA.

In Chapter 6, we investigate which topological properties bound the attainable rLFA failure case coverage in an arbitrary network, while in the second part of this chapter, we define a family of network optimization algorithms purposed at providing full rLFA coverage by augmenting the networks with a few number of suitably chosen new links.

Finally, in Chapter 7, we conclude our results and sketch possible further directions. Furthermore, the main theses of this Dissertation are summarized.

Chapter 2

Objectives and Methodology

2.1 Research Goals

The objective of this Dissertation is to give a comprehensive mathematical analysis of the different forms of LFAs and show to what extent a network can be protected by these mechanisms when single link or node failures occur. We study the aspects of a network's topology that make it unsuitable to attain full protection with LFA over that network and we also discuss best-case scenarios. Furthermore, we investigate whether the failure case coverages could be improved with only a minor changes to the topology.

- In the first part, our goal is to study the *LFA Cost Optimization* problem, which asks for manipulating the link costs to achieve higher LFA failure coverage against both single link and node failures. Besides, we carry out a complex graph theoretical analysis to understand which networks are the most suitable, how a worst-case graph looks like and what degree of density ensures full LFA protection.

- Another interesting approach is to use *LFA Graph Extension* and *LFA Cost Optimization* at the same time, since as mentioned above, in certain cases a possible negative side effect caused by one of them can be compensated by the other one. Moreover, if a network operator cannot afford to install all the additional links required to attain 100% LFA protection, he or she may alter some link costs as well to reach this end. Or in a converse case, when not all the

link costs can be tuned may an additional link can solve the problem without altering all the well-engineered pre-set link costs, which are important from the traffic engineering point of view.

- We can use rLFA instead of LFA if the routers present in the network provide this extension. Consequently, by means of complex graph theoretical analysis our aim is to identify "good" and "bad" networks for rLFA, and study to what extent does rLFA provide higher protection compared to simple LFA.

- We have seen that there are certain failure case scenarios, which cannot be protected by rLFA either. Therefore, it is important to investigate to what extent the failure case coverage can be improved by adopting any of the network optimization techniques mentioned above. In particular, our aim is to study the *rLFA Graph Extension* problem considering rare but relevant case of node failures as well.

2.2 General Assumptions

In this Dissertation, we are dealing with LFA and rLFA intended to provide fast failure recovery in an intra-domain setting. Consequently, we suppose that the analyzed network is an autonomous system, where all routers have a complete view of the network, i.e., a *link-state routing protocol* such as OSPF or IS-IS is deployed. We also assume that the calculation of next-hops is *based on only the destinations' addresses* and no other information (e.g., source routing, additional notifications) is taken into account. Moreover, *each node has a well-defined next-hop* to each destination, even if Equal Cost Multiple Paths (ECMPs) exist, i.e., when the number of shortest paths from a (source) node to a destination is more than one, then one of them is selected as the next-hop, exclusively. We further assume that only interior destinations must be protected, since in real networks, if the destination node is outside of the AS, first the corresponding egress router is found and the particular destination's address is only resolved by a second lookup[1]. Therefore, from an interior router's perspective, the protection of the egress router protects the destination itself as well. Every failure outside the autonomous system should be treated by inter-domain routing

[1]This procedure is called recursive lookup.

protocols such as BGP, or by the IGPs of other autonomous systems. Since in BGP, the routing is not solely based on shortest paths, in this Dissertation, we do not deal with failures among ASes. Furthermore, since the most common failures are transient single failures, we only deal with *single link or node failure*. As mentioned in Section 1.4, in case of multilayered networks many overlay (IP) links use one physical link as the transmission medium. The failure of such a particular link (e.g., a cable cut-off from the ground) may cause failures in a set of virtual links in the overlay topologies. Dealing such cases of multiple failures is out of scope of this Dissertation.

From graph topological aspect, we assume that the network is a *simple, undirected, weighted graph* and the links are bidirectional and point-to-point, i.e., no SRLGs or LANs (Local Area Network) exist throughout the network. Furthermore, the costs of the links are symmetric. In case of Remote LFA, we initiate the analysis in graphs with unit costs. Note that unweighted graphs are highly relevant in real-world networks (see later in Chapter 3), and as shall be shown, results for LFA can only be generalized to rLFA under the unit cost assumption.

Since an arbitrary link can only be protected if the graph of the network is *2-edge-connected*, we assume this minimum topological requirement for link-protecting case. For the case of node protection, we also assume the graph to be *2-node-connected*. Connectedness is important, since if after a failure the network fell into two distinct parts, there is no technique that can ensure a working path between an arbitrary source-destination pair.

2.3 Methodology and Notations

According to our general assumptions, we always model a network as a *simple, undirected, weighted graph* $G(V, E)$ with V being the set of nodes and E the set of links. Let $n = |V|$ and $m = |E|$, and denote the complement link set with $\overline{E}$. A particular link is denoted by (i, j), where i and j are nodes from V. Link costs are represented by a cost function $c : E \mapsto N$. The cost of a link (i, j) is denoted with $c(i, j)$. Since link costs are assumed to be symmetric $c(i, j) = c(j, i)$. In order to describe the length of the shortest path between an arbitrary node pair (u, v), we use the notation $\text{dist}(u, v)$. As a consequence of considering bidirectional and symmetric links, $\text{dist}(u, v) = \text{dist}(v, u)$.

Furthermore, $\text{LFA}_{LP}(x, y)$ is used to denote the set of nodes protecting the source-destination pair (x, y) with link-protecting LFA. $\text{LFA}_{NP}(x, y)$ marks the set of nodes protecting the source-destination pair (x, y) with node-protecting LFA. Similarly, in case of Remote LFA the set of nodes that protects source node x and destination node y with link-protecting Remote LFA is denoted by $\text{rLFA}_{LP}(x, y)$. The set of nodes that protects source x and destination y with node-protecting rLFA is indicated by $\text{rLFA}_{NP}(x, y)$. Note that the sets mentioned above are defined at node x.

Next, we show how the failure case coverage provided by LFA is measured. Given a graph $G(V, E)$ and a cost function c, let $I^{\text{LP}}_{s,d}(G, c)$ be an indicator variable whose value is 1 if node s has a link-protecting LFA to node d, and zero otherwise. Then, given a set of source-destination pairs $\mathcal{S} = \{(s_k, d_k) : k \in 1, \ldots, K, s_k \neq d_k\}$ the *link-protecting LFA coverage with respect to* $\mathcal{S}$ is defined as (inspired by [47]):

$$\eta^{\text{LP}}_{\mathcal{S}}(G, c) = \frac{1}{|\mathcal{S}|} \sum_{(s,d)\in\mathcal{S}} I^{\text{LP}}_{s,d}(G, c) \ . \tag{2.1}$$

Similarly, let $I^{\text{NP}}_{s,d}(G, c)$ be an indicator variable for node-protecting LFA coverage[2]. Note, however, that in case of node protection special care must be taken to handle the so called *last-hop problem*, which arises when d is an immediate neighbor of s and the default shortest path between them is exactly the link (s, d) (see, for instance, the case of the source-destination pair (b, d'') in Fig. 1.5). In such cases, the node failure we want to protect is exactly the failure of the destination d itself, which case is hardly protectable by LFA. Therefore, to resolve this issue and increase the overall level of protection, it is recommended to support fallback to link protection in these cases [90]. However, from a traffic engineering point of view maximizing the coverage in such a way may not be the right approach, since the affected traffic may be dropped on congested backup links. So, there is no standard on how this case should be treated.

Therefore, in this Dissertation we consider both cases. During our LFA analysis, for such source-destination pairs we only require that the link (s, d) be protected by a link-protecting LFA, and we ignore the node-protection requirement. Consequently, $I^{\text{NP}}_{s,d}(G, c)$ takes the value 1 if and only if

[2]Note that, according to the objective, LFA coverage can be calculated differently, for instance, by the fraction of the traffic that is lost due to missing LFA.

(i) d is not the immediate next-hop of s to d and s has a node-protecting LFA to d, or

(ii) d is the immediate next-hop of s to d and s has a link-protecting LFA to d.

Then, the node-protecting LFA coverage is defined as

$$\eta_{\mathcal{S}}^{\mathrm{NP}}(G,c) = \frac{1}{|\mathcal{S}|} \sum_{(s,d)\in\mathcal{S}} I_{s,d}^{\mathrm{NP}}(G,c) \ . \tag{2.2}$$

On the other hand, in our Remote LFA analysis we consider node protection as *undefined* between an arbitrary neighboring node pair. However, if there is a significant difference between the two approaches, we shall examine both cases.

In some cases, it will be convenient to refer to both link-protecting and node-protecting LFAs under a common term. In such cases, we shall only say "an LFA exists" and the corresponding coverage metric will be written as $\eta_{\mathcal{S}}(G,c)$. Moreover, we often confine ourselves to the special cases when $\mathcal{S}$ is the set of all node pairs whose destination is a given terminal node d: $\mathcal{S}_d = \{(s,d) : s \in V \setminus \{d\}\}$, or when $\mathcal{S}$ contains all distinct node pairs in $V \times V$. In the latter case, we neglect to indicate $\mathcal{S}$ in the LFA coverage metric and simply write $\eta^{\mathrm{LP}}(G,c)$, $\eta^{\mathrm{NP}}(G,c)$, and we use the shorthand notation $\eta(G,c)$ when LFA type does not matter.

In case of Remote LFA, the corresponding link-protecting coverage $\mu_{\mathcal{S}}^{LP}(G,c)$ and node-protecting coverage $\mu_{\mathcal{S}}^{NP}(G,c)$ are calculated similarly. However, during our Remote LFA analysis we always assumed that $\mathcal{S}$ consists of all distinct node pairs in $V \times V$, therefore denoting $\mathcal{S}$ was always neglected. Furthermore, since we consider only unit cost networks, we also disregard to denote cost function c and simply use the notations $\mu^{LP}(G)$ and $\mu^{NP}(G)$, respectively.

With this in mind, a simplified metric for measuring rLFA coverage can be considered as follows:

$$\mu_{LP}(G) = \frac{\#\mathrm{rLFA_{LP}\ protected}\ (s,d)\ \mathrm{pairs}}{\#\mathrm{all}\ (s,d)\ \mathrm{pairs}} \tag{2.3}$$

$$\mu_{NP}(G) = \frac{\#\mathrm{rLFA_{NP}\ protected}\ (s,d)\ \mathrm{pairs}}{\#\mathrm{all\ non\text{-}adjacent}\ (s,d)\ \mathrm{pairs}} \tag{2.4}$$

Similarly to LFA, when rLFA type does not matter, we use the shorthand notation

$\mu(G)$. Further notations are explained before their first usage.

In our model, we always sought graph-theoretical *lower and upper bounds* on failure case coverages in order to study the performance of the different protection mechanisms in artificial and in real-world network topologies. In the former case, we generated different networks known from the literature such as topologies used in telecommunications (e.g., rings, grids) and well-known topologies from graph theory such as Möbius ladder and bipartite graphs.

On the other hand, real-world topologies could be inferred from several existing ISP (Internet Service Provides) datasets such as the Rocektfuel [91], SNDLib [92], and Topology Zoo [93]. From the Rocketfuel dataset, we used AS1221, AS1239, AS1755, AS3257, AS3967 and AS6461. We obtained Point of Presence level (POP) maps by collapsing the topologies so that nodes correspond to cities and we eliminated leaf-nodes (this preprocessing method was suggested in [94]). These networks come with inferred link costs (these costs are needed to compute the "default" LFA coverage $\eta(G, c)$ of the network). We also chose some network topologies from SNDLib, namely, the Abilene, Italy, Germany, NSF and AT&T networks and the 50 node extended German backbone (Germ_50). Unfortunately, except for the last network no valid link costs were available, so we set each cost to 1. Finally, some representative ISP topologies from Topology Zoo were used as well, in particular, the Arnes, Deltacom, Geant, Gambia, and the InternetMCI topologies. For these networks, only link bandwidths were available, therefore, according to Cisco's recommendation, we set all link costs inversely proportional to link capacities. Otherwise, we assumed that link costs are unit costs or we used randomly generated ones[3]. The topologies used in this Dissertation are described in Table 2.1.

We were always careful to study networks with different topological properties, e.g., size of the network, density, average node degree, connectivity. After we found that in an arbitrary network the failure case coverages can be particularly poor, we tried to find *network optimization methods* to improve it. In these cases, our aim was solely to *improve the failure case coverages* of the different forms of LFAs, and we did not deal with traffic engineering or load balancing related issues, or possible congestion that could occur when the protected traffic is directed to a (possibly) crowded link. Readers interested how these questions should be addressed are referred to [90]. To

[3]In each case of our analysis, the chosen cost function will be noticed.

Table 2.1. Inferred real-world topologies. The number of nodes and the number of links are denoted by n and m, respectively

	Name	n	m
Rocketfuel	AS1221	7	9
	AS1239	30	69
	AS1755	18	33
	AS3257	27	64
	AS3967	21	36
	AS6461	17	37
SNDLib	Abilene	12	15
	AT&T	22	38
	Germ_50	50	88
	Germany	17	25
	Italy	33	56
	NSF	26	43
TopologyZoo	Arnes	41	57
	Deltacom	113	161
	Gambia	28	28
	Geant	37	55
	Interoute	19	33

validate proposed theorems mathematical proofs were made. In general, the results indicated that *most of the problems are NP-complete*, so *exact algorithms (ILPs)* were formulated to solve them. However, in most of the cases (algorithms mainly run on real network topologies), *approximating heuristics* were necessary to obtain reasonable results within a limited time frame. In the latter cases, we used *greedy algorithm* as a first approach. Then, a *simulated annealing based heuristic framework* was developed with many *tunable parameters* to avoid getting stuck in local optima that may happen by the greedy approach. However, as shall be shown, the greedy approach outperformed the simulated annealing based framework in many cases.

Last but not least, the simulation tools were implemented particularly in C++/ LEMON[4] and the results were conducted through various BASH[5] scripts. In some

[4]LEMON stands for Library for Efficient Modeling and Optimization in Networks. It is a C++ template library providing efficient implementations of common data structures and algorithms with focus on combinatorial optimization tasks connected mainly with graphs and networks (http://lemon.cs.elte.hu/trac/lemon accessed in October 2013).

[5]Bourne Again SHell - it is a command-line interface for interacting with Unix/Linux based operating systems.

cases, the networks were described in Geographic Markup Language (GML), therefore in order to analyze them with LEMON a conversion was needed, which was carried out by our own software, called GML2LGF converter [95].

Chapter 3

Related Work

As observed above, the strength of LFA lies in its simplicity, however, its dependence on the underlying network topology has a huge impact on its performance. In particular, extensive simulations and numerical studies [96, 82, 97, 78, 83] have shown that LFA can only protect $75 - 85\%$ of the link failures and $50 - 75\%$ of the node failures, respectively. Recently, there have been various attempts to deeply analyze LFA in different networks in order to find the topological properties that bound the LFA coverage and to alleviate this drawback by optimizing the topology instead of modifying the technique itself to achieve higher failure case coverage.

With this in mind, first we discuss the lower and upper bounds attainable by LFA. Then, we show different network optimization techniques aimed at increasing LFA coverage by augmenting the network with new links. Since one of our main goals is to optimize the link costs to reach the same end, we demonstrate in which fields this approach has already been proven to be successful to achieve certain goals. Finally, we discuss the role of setting the IGP link costs in operational networks concentrating on the case of the unit cost setting, a general assumption we shall make in Chapter 5. Note that in this section, besides the summarized related work, each detailed propositions and definitions are also referred from the literature, thus they are not the results of this Dissertation.

3.1 Lower Bounds on LFA Coverage

In order to identify worst case graphs for LFA, in [84] it was found that from a graph topological aspect even rings have the smallest LFA coverage out of all 2-connected graphs with the same number of nodes. Moreover, the authors found the following lower bound of LFA failure case coverage of a 2-connected graph measured by $\eta(G)$.

Proposition 1. *The LFA failure coverage of a 2-connected graph G on n nodes is bounded by $\frac{1}{n-1} \leq \eta(G) \leq 1$ and the lower bound is tight for rings with even number of nodes and uniform edge costs.*

Next, we show the failure case coverage of an odd ring with unit link costs.

Proposition 2. *For an odd ring with $n > 2$ and uniform costs: $\eta(G) = \frac{2}{n-1}$.*

On the other hand, it was observed that an undirected, simple graph with uniform link costs is fully protected by LFA if and only if each link is contained in at least one triangle (cycle of length 3). They also showed that in case of arbitrary link costs full LFA coverage can only be provided if and only if all next-hops remain reachable after a link failure.

In [98], *Tapolcai* studied to what extent the average node degree leverages the LFA failure coverage in an arbitrary network. In particular, he has given tight graph theoretical lower and upper bounds on the LFA coverage achievable in a given graph under *any* selection of link costs. The bounds are based on the fact that a shortest path tree to some destination d can contain only $n - 1$ links, and all the remaining links can be used for providing LFAs to their endpoints.

Proposition 3. *For any connected simple graph G with $n > 2$:*

$$\eta^{NP}(G, c) \leq \eta^{LP}(G, c) \leq \frac{n}{n-1}(\Delta - 2) + \frac{2}{n-1} \ .$$

What the above proposition in essence says is that in large sparse graphs LFA-coverage is upper bounded by the average node degree: $\eta(G, c) \leq \Delta - 2$. The next proposition gives a lower bound on the LFA coverage. However, the result concerns link-protecting LFAs exclusively.

Proposition 4. *For any connected simple graph G with $n > 2$:*

$$\eta^{LP}(G,c) \geq \frac{n}{n-1}\frac{\frac{\Delta}{2}-1}{\Delta_{\max}-1} + \frac{1}{(n-1)(\Delta_{\max}-1)} \ .$$

The most important message of these propositions is that LFA coverage increases with the average node degree Δ, that is, the denser the network the higher the link-protecting LFA coverage. This raises the question whether we can find graphs of low degree with 100% LFA coverage. *Tapolcai* found that a 2-connected graph with the smallest possible average degree that can still be fully protected using LFA is the 3-cycle. Every other 2-connected graph with complete LFA coverage has an average degree higher than 2. Furthermore, graphs with $\Delta < 2$ cannot have full protection because such graphs contain at least one node with degree 1 whose single outgoing link can never be protected.

3.2 Optimizing Network Topology to Improve LFA Coverage

Besides the study of how to design LFA-friendly network right from the outset [83], a deeper analysis was made in [84]. It turned out that in an already deployed operational network full LFA protection can only be achieved at the cost of a substantial topology redesign. Therefore, as a way of improvement *LFA Graph Extension* problem was studied, which asks to extend the network with the fewest number of new links to improve LFA coverage. In particular, in case of uniform link costs the problem (called *minLFAu*) was stated as follows:

Definition 1. *Given a simple, undirected graph $G(V,E)$ with uniform link costs c on all links and an integer l, is there a set $F \subseteq \overline{E}$ with $|F| \leq l$ and $\forall (u,v) \in F$: $c(u,v) = c$ so that $\eta(G(V, E \cup F)) = 1$?*

Note that in case of uniform link costs an additional link alters at least one shortest path, since nodes connected by the new link will use it to reach each other.

In case of arbitrary weighted graphs, the problem *minLFAw* was similarly proposed except that the cost of the new link should be properly chosen in order to avoid

the alteration of the existing shortest paths[1]. It turned out that LFA Graph Extension problem is NP-complete in both cases. Therefore, an optimal algorithm (ILP) and an approximating greedy heuristic were developed. Although, the ILP was proven intractable in larger topologies. It was shown that approximately 95% LFA failure case coverage can be realized by adding no more than $2-4$ new links. Moreover, the first iteration (i.e., after the first new link is added) attains a significant improvement ($5-12\%$). This led to a separate problem, called *LFA Graph Improvement*, where the task is merely to raise LFA failure case coverage instead of boosting it to 100% protection.

The authors of [99] extended the aforementioned problems from the single link failure model to the case of node failures as well, and what is more, ECMPs and broadcast LANs were also taken into account. Furthermore, in order to find better approximating algorithm various well-known heuristics (LJC [100], SBT [101], RSBT and MSBT from [102] and a backtracking algorithm) available from the literature were studied. During their node protection analysis, it was found that dozens of new links are required to achieve 100% failure case coverage, which was a clear indication to rely on LFA Graph Improvement instead in order to keep the number of new links small. The most important observation was that on average, using MSBT algorithm provided the highest LFA coverage with the smallest number of new links, however LJC improved the LFA coverage the most in the initial steps. In particular, adding only 5 new links resulted in $10-15\%$ improvement in failure case coverage, while the next 5 links boosted LFA protection to $90-95\%$.

In a subsequent study [103], the aforementioned framework was further improved to handle the case of Shared Risk Groups as well. It was observed that as SRG density[2] increases the initial LFA coverage drops drastically. In particular, when SRG density is only 10% the LFA coverage is decreased merely by $2-6\%$ compared to single-failure case. However, if the density is around 50%, then the difference increases to $20-25\%$ for initially better protected networks, while this number is 10% in case of less protected ones. In case of 90% SRG density, the initial link-protecting LFA coverage falls to about $10-20\%$, and $3-12\%$ for the node-protecting case. Surprisingly, it was found that the number of required new links does not grow

[1]The new cost should be larger than the length of the longest shortest path.
[2]SRG density denotes the fraction of all adjacent dual-link sets to be selected as local SRGs.

at a similar pace as in case of 10% SRG density, since only $2-3$ more links were necessary. In case of 50% SRG density, about $3-4$ more links were necessary, while in case of 90% SRG density about $7-12$ more links are needed. This essentially means that even if 90% SRG density is present in a network, we still can improve the link-protecting LFA coverage to 100%. However, it must be noted that the newly added links do not form and do not become a part of any SRGs.

3.3 Link Cost Optimization

Besides augmenting a network with new links to increase the LFA coverage, we can alter the link costs instead. Studying this approach is one of the main aims of this Dissertation, which is presented later in Chapter 4. As a matter of fact, this process does not require any modifications to the underlying routing protocols as changing a link metric has always been a feature of IGPs. Furthermore, as discussed above, choosing a link cost properly also became an important issue when a weighted network is being augmented.

However, changing link costs is a useful technique during other pre-planned maintenance operations, for instance, when a link is desired to be shut down. As mentioned in Section 1.1.1, IGPs' re-convergence processes can lead to temporary micro-loops, which should be avoided.

3.3.1 Avoiding Micro-Loops

Below, we discuss how the cost of the link desired to be shut down should be altered by never letting the routers in an inconsistent forwarding state during the re-convergence process. Before manually shutting down a link, operators can collect measurements of the ongoing traffic and use traffic-engineering tools to predict the possible effects of changing a particular link cost [104]. However, it was found that this problem is NP-hard, even for the simplest objective functions, therefore simpler approaches should be investigated.

Teixeira *et al.* [105] advised that the operators prepare the network for the impending topology change by increasing the link cost to a very high value. This method has two main advantages; on the one hand, there is no need for failure detection as

adjacent routers are informed immediately and, on the other hand, the link can continue to carry packets in flight, while the routers converge to the new topology. After the network converged, operators can safely disable the link. The inverse procedure can also be used before adding a new link to the network, since operators can test whether the link works properly before routers start to use it for forwarding. However, if the routers directly connected by the link desired to be shut down refresh their FIBs at different times, then forwarding loops can still occur.

Shand *et al.* [106] discussed the idea of repeatedly incrementing a link cost by one to reach a forwarding state, where the link is no longer used[3]. This solution was rejected by the IETF, since the number of increments required for such a link can be very high due to the wide range of costs that are used in an operational network.

To overcome this issue, a solution was proposed in [108], which only performs as many link cost updates as it is necessary to avoid forwarding loops. The authors found that with their technique, across all the investigated topologies, 85% of the links can be shut down with less than three metric changes. Thanks to the improvements proposed in a subsequent study [109], it is now possible to compute the metric sequence directly on the routers. Their evaluations showed that in most networks, the metric sequence for a link shutdown can be computed in a few tens or hundreds of milliseconds.

3.3.2 Achieving Traffic Engineering Goals

Optimizing link costs proved to be useful to achieve further traffic engineering goals as well. Since in intra-domain routing the shortest paths are determined by the administrative link costs, it is often the case that ECMPs exist between a certain source-destination pair. In such cases, routers distribute traffic over all available equal cost paths. However, when single shortest path routing (SSP) is required, i.e., one next-hop must be selected exclusively, choosing a particular path among the available shortest paths is based on a so called tie-breaker. Unfortunately, tie-breakers may use non-deterministic information such as interface or port numbers to reach this end making the paths of general SSP routing hard to predict. From a traffic engineering point of view, however, it would be necessary to know which

[3]Later, this work became an informational standard (RFC 5715, [107]) and it provides a survey of the currently proposed mechanisms.

packets are carried by which links. Note that in case of failures ECMPs could exist around the failed component. Therefore, not just primary paths, but backup paths also need to be predictable. In order to resolve this issue, shortest paths need to be unique throughout the network, this way enabling deterministic path selection and evading the need for tie-breakers. To provide such unique shortest path routing (USP, [110, 111]), a proper link cost setting is required.

In [110, 111], these problems were investigated under the assumption that Not-Via is deployed as an FRR method, and it was shown that when link costs are optimized for wrong tie-breakers, routers possibly send traffic to already highly loaded links. In case of USP routing, first 1000 random link cost settings were generated to quantify the USP probability. The results indicated that the maximum size of the interval $[1, k_{max}]$, from which cost values can be assigned to links, has a significant impact on USP probability. For instance, in the analyzed Tiscali network, wherein 38 nodes are connected by 232 links, with $k_{max} = 2^{10}$ the USP probability was below 10% for resilient routing and 50% for failure-free USP routing. Inspired by [112], therefore, routing optimization for USP was carried out. It was found that for $k_{max} \in [4 : 7]$ in failure-free case and for $k_{max} \in [9 : 20]$ in resilient routing the proposed heuristic was successful in finding USP solutions. For small k_{max}, the found USP solutions always led to high link utilization.

In a subsequent brief study [113], it was stated that considering Loop-Free Alternates as the deployed FRR method, optimizing link costs to achieve higher coverage can lead to extremely high link utilization. Considering link utilization as a secondary objective still does not alleviate this drawback. However, if the order of objectives are swapped, then low utilization can be achieved with relatively high coverage. Nevertheless, these problems in [113] were not defined formally, neither their complexity and comprehensive numerical evaluations were presented.

In [114], another traffic engineering technique was proposed, where link-protecting LFA coverage was also taken into account. However, the algorithm was evaluated only in one real-world network, namely in Abilene network [115], where even if very low link utilization could be achieved, the link-protecting LFA coverage was only $\sim 50\%$. Higher failure case coverages with lower link utilization could only be reached in synthetic topologies generated by iGEN [116].

Nonetheless, we have no information about whether 100% LFA protection could

ever be reached in an operational network by merely optimizing link costs.

3.4 Unit Cost Networks

The efficiency of LFA in protecting most failure scenarios crucially depends on both the graph topology and the link costs of the underlying network. Unfortunately, it was found in [84] that it is extremely difficult to consider both at the same time, due to the complexity of the related graph theoretical questions. Therefore, it has proven beneficial to first study graph topological concerns separately from the effects of link costs. During our Remote LFA analysis, we follow the same course as the first approach.

On the other hand, considering unweighted graphs is fruitful for a number of further reasons, for instance, this case is highly relevant in real-world networks. In particular, despite the fact that OSPF has an option for setting the link costs dynamically, uniform link costs throughout an AS can easily be realized when all links have the same bandwidth[4]. Moreover, OSPF assigns a default cost metric of 1 to any link faster than 100 Mbps [117, 118][5].

In IS-IS, there is no option for dynamic cost setting, thus without manually changing the link costs, to all links the same metric of 10 is assigned by default [120]. However, some recent router firmwares have started to support dynamic cost setting option in IS-IS in order to optimize routing[6].

On the other hand, as shall be shown in our Remote LFA analysis, considering uniform link costs is useful to generalize earlier results for LFA to rLFA.

[4]The formula to calculate the cost is the reference bandwidth (100 Mbps) divided by the interface bandwidth.

[5]In order to change the reference bandwidth in Cisco routers, see `auto-cost reference-bandwidth` command in [119].

[6]For details, see the `reference-bandwidth` option in [121].

Chapter 4

Improving LFA Coverage by Optimizing IGP Link Costs

In this chapter, we discuss in detail how the basic Loop-Free Alternates [47] works, we show its different protection schemes and its failure case coverages that can be attained in generated and real-world network topologies. After revealing its advantages and disadvantages, we study LFA Cost Optimization problem, that is, the task of improving LFA coverage by tuning IGP link costs. Finally, we examine to what extent the failure case coverage can be improved further by unifying LFA Graph Extension and LFA Cost Optimization.

4.1 Loop-Free Alternates in Detail

As it was briefly shown in Section 1.5, in LFA, when a failure occurs, the adjacent router tries to pass the packet to an alternate neighbor that still has a functioning path to the destination.

Next, we discuss the conditions and different protection schemes of LFA formally. Consider the network depicted in Fig. 4.1, where all links have unit costs except link (n, e). Suppose that node s wants to send a packet to node d and its default next-hop e is unreachable, since the link (s, e) between them went down. In this case, s has to find an alternate neighbor, which will not pass the packet back. Fortunately, node n fulfills this requirement, so s can reroute the traffic destined to d towards n. Here, we say that n is a link-protecting LFA for node s towards destination node d [47].

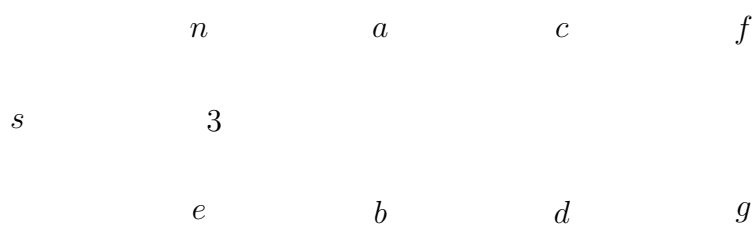


Figure 4.1. A sample network topology, where the shortest paths from s to d, and from n to d are marked by arrows

Definition 2. *Given a connected graph $G(V, E)$, some source s and destination d, let e be the default next-hop of s towards d. Then, some neighbor n of s is a* link-protecting LFA *for s to d if*

(i) $n \neq e$, *and*

(ii) *the loop-free condition applies:*

$$\text{dist}(n, d) < \text{dist}(n, s) + \text{dist}(s, d) \ . \tag{4.1}$$

One can easily verify this condition. If the distance from node n to d is strictly less than the distance from n back to the source node s then to node d means that node n will not pass the packet back to s, this way bypassing the failed link (s, e). Recall that $\text{dist}(x, y)$ denotes the shortest path distance from node x to node y. In other words, any neighbor that is not an upstream in the shortest path tree rooted at d is a link-protecting LFA.

Next, consider that node d is the source, and node g is the destination. Then, under the assumption that the link (d, g) has failed, d has no loop-free alternate, since passing the packets to node c may cause loop as node c does not know about the failure and its default next-hop to node g may be node d[1] (i.e., condition 4.1 does not apply). Considering node d's other neighbor, node b, is still not an LFA, since from the aspect of node d, node b is an upstream node in the shortest path tree.

Similarly to link protection, node-protecting LFA can also be defined. Suppose again that in Fig. 4.1, node s wishes to send a packet to node d. If not just the link (s, e), but the next-hop e itself fails, then passing the packets towards node n would

[1]Since node c has ECMP to node g.

avoid node e, since n's shortest path to node d goes towards node a. In this case, we say that node n is a node-protecting LFA for node s towards destination node d [47].

Definition 3. *For some source s and destination d, let e be the default next-hop of s towards d. Then, some neighbor n of s is a* node-protecting LFA *for s to d if, in addition to* (i) *and* (ii) *in Definition 2, the node-protection condition also applies:*

$$\text{dist}(n, d) < \text{dist}(n, e) + \text{dist}(e, d) \ . \tag{4.2}$$

However, if the cost of link (n, e) were 1, then node n would be solely a link protecting LFA for the traffic originated at node s and destined to node d, since it would have ECMP towards node d (i.e., condition 4.2 does not apply).

Note that as mentioned in Section 2.3, in case of node protection the last-hop problem can be handled differently. We can assume that (i) the type of protection between neighboring source-destination pairs support *fallback to link-protecting LFA* instead; (ii) or the protection is considered as *undefined*. In this Chapter, we assume (i), while later, in our Remote LFA analysis, we consider (ii). In certain cases, when choosing a particular policy results in significant difference than the other, we shall investigate both cases.

Collectively, a link-protecting or node-protecting LFA for a given destination is termed *per-prefix LFA*, since they are protecting a particular prefix. According only to the case of link protection, *per-link LFA* can be defined as an alternate next-hop, which protects every destination originally reached through the failed link, i.e., a per-link LFA n, is a per-prefix LFA for all the destinations reached through that failed link. Note that *per-node LFA* can be similarly defined. In our first example, node n is a per-link LFA with respect to the failure of link (s, e) for destinations e,b,d,g, but it is per-node LFA with respect to the failure of node e, only for destinations d and g.

Next, we show that an LFA, which is solely link-protecting (by Definition 4.1) in certain cases can protect against the failure of the next-hop node as well. Consider Fig. 4.2 and suppose that node s wishes to send packets to node d and suddenly next-hop e fails. Now, passing the packet to, say node n, would not be a good solution, since it has got ECMP to node b and it may would use the path, which goes through node e. However, since n is adjacent to node e, it will also recognize the failure of node e, and it will pass the packets to its LFA (to node a). In this case, n is a *de*

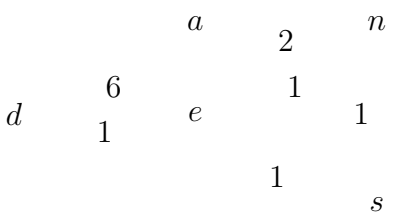


Figure 4.2. An example when de facto node-protecting causes micro-loop

facto node-protecting LFA. Note that relying on de facto node-protecting LFAs may give rise to LFA loops, since the LFA of a source's neighbor may be the source itself. In particular, when the two neighboring nodes s and n have the same next-hop e for destination d, and next-hop e went down, then s would pass the packet to n, and n would pass the packet back, since they are link-protecting LFAs of each other (indicated by the gray arrows).

4.2 Problem Formulation

As discussed in Section 1.6, there are several ways to improve the LFA coverage by different network optimization techniques. First, in this section we investigate *LFA Cost Optimization problem*, which again asks to optimize link costs in a way as to maximize the level of protection provided by LFA.

Formally, *LFA Cost Optimization problem for the link-protecting case* can be defined as follows:

Definition 4. *LFACostOptLP(G, $\mathcal{S}$): Given a graph $G(V, E)$ and a set of source-destination pairs $\mathcal{S}$, is there a cost function c so that $\eta_{\mathcal{S}}^{LP}(G, c) = 1$?*

Easily, a similar problem formulation LFACostOptNP(G, $\mathcal{S}$) exists for the node-protecting case as well. When no ambiguity arises, we shall refer to both problems simply as LFACostOpt. In addition, we shall in many cases treat the optimization version of these problems, that is, we shall seek the costs that maximize network-wide LFA coverage instead of merely asking whether or not a cost setting for full protection exists.

4.3 LFA Cost Optimization

Next, we turn to the LFA Cost Optimization problem. First, we characterize the extent to which such an optimization can improve LFA coverage, then we discuss the complexity and the algorithmic aspects of the problem. Most of the observations apply to LFAs generally, regardless of link protection or node protection, so, unless otherwise stated, the term LFA will refer to link-protecting LFAs in the sequel. We shall indicate clearly in the text when LFA types indeed matter.

4.3.1 The Potential of LFA Cost Optimization

The question immediately arises as to whether it is worth optimizing costs for LFA at all. In Section 1.6, we showed through an example that altering link costs can produce LFAs to protect certain source-destination pairs, which were unprotected otherwise. However, readjusting costs in most of the cases changes, possibly in a negative way, default shortest paths, which might have been previously tweaked with great accuracy to match the needs of the network in terms of load balancing, traffic engineering, etc. [122, 123, 124]. On the other hand, as shall be shown below, the wins achievable with optimizing link costs for LFA can be substantial, and such a huge improvement in fast resiliency might compensate for the losses in forwarding efficiency in certain cases. In particular, we show that LFA Cost Optimization can improve the link-protecting LFA coverage by more than 50% in certain cases.

Theorem 1. *For any $k > 0$ integer, there is a graph G on $n = 4k+2$ nodes with two cost functions c_1 and c_2, so that $|\eta(G, c_1) - \eta(G, c_2)| \geq \frac{1}{2}$.*

Proof. First, consider the so called "Möbius ladder" topology with unit costs depicted in Fig. 4.3(a), wherein the link-protecting LFA failure coverage $\eta^{\mathrm{LP}}(G, c) = 0.4$. However, in Fig. 4.3(b), the costs of diagonals are chosen so that the paths between any two nodes are shorter around the ring than through it via a diagonal. This way, as one easily verifies, every source-destination pair is protected, i.e., $\eta^{\mathrm{LP}}(G, c) = 1$. Moreover, since the paths from the LFAs reached by the diagonals bypass the failed next-hops as well in every case, $\eta^{\mathrm{NP}}(G, c) = 1$.

This graph construction can be generalized to arbitrary $n = 4k + 2$, where $k \in \{1, 2, 3 \ldots\}$, and one can always choose the above cost setting strategy to achieve

(a) Wheel graph with unit cost links

(b) Wheel graph with optimized link costs

Figure 4.3. Illustration for Theorem 1

complete LFA protection. For instance, consider the Möbius ladder topology depicted in Fig. 4.4, which is a similar wheel graph on $n = 10$ nodes but drawn in a slightly awkward layout, and all link costs are uniformly set to 1. The layout was chosen so that one can easily check the validity of the following claim for any Möbius ladder with $\frac{n}{2}$ odd[2], $n > 2$ and c uniform: for every $d \in V$, exactly $\frac{n}{2}-1$ nodes have link-protecting and node-protecting LFA to d. Considering node d we marked in Fig. 4.4, there is exactly one node in each "column" that has an LFA to d, except for the column of d in which there is no protected node. In this case, the LFA failure case coverage for this network G with cost setting c, $\eta(G,c) = \frac{n(\frac{n}{2}-1)}{n(n-1)} = \frac{1}{2}\frac{n-2}{n-1} = \frac{1}{2}(1-\frac{1}{n-1}) = \frac{1}{2}-\frac{1}{2}\frac{1}{n-1} < \frac{1}{2}$, again, in terms of both link-protection and node-protection. For instance, in our example $\eta(G,c) = \frac{4}{9}$. However, if the costs of the diagonals (links in the columns) are chosen to $\frac{n}{2}+1$, then $\eta(G,c) = 1$. □

Note that for any Möbius ladder G with $\frac{n}{2}$ even, $n > 8$ and c_1 uniform the following claim applies: for every $d \in V$, exactly $\frac{n}{2}$ nodes have LFAs to d. Thus, if a cost function c_2 sets the costs the diagonals as mentioned above, $|\eta(G,c_1) - \eta(G,c_2)| < \frac{1}{2}$, since the LFA failure case coverage for G with cost setting c_1, $\frac{1}{2} < \eta(G,c_1) = \frac{\frac{n}{2}}{n-1} = \frac{1}{2}+\frac{1}{2}\frac{1}{n-1} \leq \frac{6}{11} < \frac{3}{5}$.

This example shows that the different selections of link costs can produce dramatic differences in LFA failure case coverage. Simulation studies presented later also seem to support this claim. The other lesson is that resilience and forwarding efficiency are usually contradicting requirements in routing: in our example, in the latter case, all traffic flows along min-hop paths but resilience is poor, while in the former case, we have full protection but long forwarding paths going around the ring instead of

[2]Note that $\frac{n}{2}$ odd means that $n = 4k+2$.

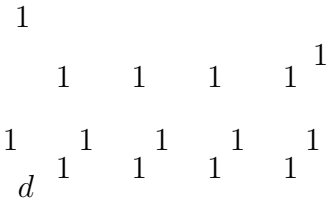


Figure 4.4. A Möbius ladder topology, where $\eta(G, c) < \frac{1}{2}$

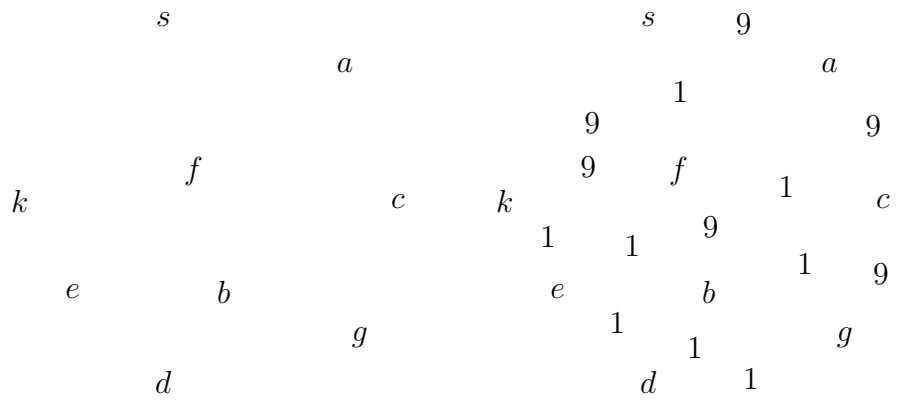
taking the shortcuts through it. Such "joker" links that do not carry traffic seem a general requirement for protectability [125].

4.3.2 Computational Complexity

Next, we show that the full-fledged LFA Cost Optimization problem given in Definition 4 is NP-complete. This result is not particularly unexpected, as it was found that basically all other LFA-related network optimization problems are NP-complete as well [84]. Taking a closer look, we find that there are two reasons due to which the problem is difficult. First, there is an inherent coupling between the LFAs to different destinations through the link costs, which makes it difficult to take independent decisions. In particular, assigning a neighbor as an LFA towards some destination necessitates adjusting link costs accordingly, but this may destroy LFAs to other destinations. Second, even assigning LFAs to *just a single destination* seems difficult enough. For proving the NP-completeness, we reduce an already known problem, called Protection Routing ($PR(G, d)$), proved to be NP-complete [72], to the LFA Cost Optimization problem.

Definition 5. *PR(G, d): given a graph $G(V, E)$ and some $d \in V$, is there a directed spanning DAG $R_d(V, E_d) : E_d \subseteq E$ rooted at d, so that for any single node or link failure f every node $s \in V \setminus \{d\}$ has a neighbor $k : (s, k) \notin E_d$ for which it holds that* (i) *k is not upstream of s in R_d^f, and* (ii) *there is a $k \to d$ path in R_d^f, where R_d^f is obtained from R_d by removing the failed component f.*

A simple use-case scenario of PR(G, d) is depicted in Fig. 4.5(a), where the spanning DAG $R_d(V, E_d)$ is rooted at d and is denoted with solid black arrows. The gray links indicate all the remaining links that do not belong to $R_d(V, E_d)$. Node f in the DAG marks the failed node, while the thicker gray link marks the neighboring



(a) A given spanning DAG rooted at d in the PR(G, d) problem

(b) A certain cost function that results in the same DAG

Figure 4.5. An illustration for Theorem 2

link $(s, k) \notin E_d$ between node s and its neighbor k. Now, s is protected against the failure f according to Definition 5, as in this case by (i) all $s \to d$ paths within R_d towards d does not go through node s, and by (ii) in R_d^f, i.e., in the DAG, where the failed component f and all its links are removed, k still has a path towards d without violating (i).

First, consider the following theorem characterizing the difficulty of the *node-protecting case* of LFA Cost Optimization problem.

Theorem 2. *Given a graph $G(V, E)$ and a node $d \in V$, LFACostOptNP(G, $\mathcal{S}_d$) with $\mathcal{S}_d = \{(s, d) : s \in V \setminus \{d\}\}$ is NP-complete.*

Proof. Easily, LFACostOptNP(G, $\mathcal{S}_d$) is in NP. To prove NP-hardness, we show that it is essentially equivalent to PR(G, d).

The basic differences are that *(a)* LFACostOpt(G, $\mathcal{S}_d$) is defined in terms of costs, i.e., routing is based on shortest paths, while PR(G, d) in terms of a routing DAG R_d, i.e., paths are not restricted to shortest paths, and *(b)* item *(ii)* in the above definition. To show equivalence, we need to handle these differences.

First, we show that a cost function c uniquely determines R_d and vice versa (see Fig. 4.5(b)) in that we can show a mapping from c to R_d so that a path is shortest path over c if and only if it is contained in R_d (this will handle *(a)*). Easily, the shortest paths over c are always in a DAG. The reverse direction, that is, taking R_d and creating a cost c of it, is equally easy: take a topological ordering $o(v) : v \in V$

of R_d (this always exists) and for each $(i, j) \in E$ set $c(i, j) = o(j) - o(i)$ if $(i, j) \in E_d$ and $c(i, j) = n$ otherwise.

Second, *(b)* means that in PR(G, d) we only take a node for protected, if after a failure f all the paths of the the secondary next-hop in R_d avoid f. However, this is guaranteed by the node-protecting condition (4.2). □

A similar result can be shown for the *link-protecting case* as well. Below, we only state the result but we do not give a detailed proof. We only note that the proof involves observing that the NP-completeness argumentation in [72] remains valid if we treat link failures only and disregard node failures, from which the derivation is straight forward.

Theorem 3. *Given a graph $G(V, E)$ and a node $d \in V$, LFACostOptLP(G, $\mathcal{S}_d$) is NP-complete.*

Theorem 2 and Theorem 3, stated for the special case of LFACostOpt(G, $\mathcal{S}_d$), yield the following complexity characterization for the full-fledged LFA Cost Optimization problem.

Corollary 1. *The LFA Cost Optimization problems LFACostOptLP(G, $\mathcal{S}$) and LFACostOptNP(G, $\mathcal{S}$) are NP-complete.*

Additionally, we also observe that the optimization version, which asks for a cost maximizing LFA coverage, is also intractable.

4.3.3 An Exact Algorithm

LFA Cost Optimization is difficult, yet solving it would be extremely useful for improving the resilience in operational IP networks. Next, we give an Integer Linear Program (ILP) suitable for obtaining optimal solutions only in small networks. For simplicity, we assume that $\mathcal{S}$ contains all distinct node-pairs, noting that the algorithms are easy to generalize to arbitrary $\mathcal{S}$.

The ILP is formulated in the dual space: to every node i we assign a node potential π_i^d that signifies the shortest distance from i to some d over the costs c, and then we require that the potentials and the costs together fulfill the Shortest Path Optimality Criteria [126] while also maximizing LFA coverage.

Consider the below ILP for the link-protecting version of the LFA cost optimization problem:

$$\max \sum_{(s,d)\in S} \alpha_s^d \tag{4.3}$$

$$\pi_j^d + s_{ij}^d = \pi_i^d + c_{ij}, \quad 0 \le s_{ij}^d \le C y_{ij}^d \qquad \forall (s,d) \in \mathcal{S}, \forall (i,j) \in E \tag{4.4}$$

$$\sum_{v\in N_s} y_{sv}^d \le |N_s| - 1 \qquad \forall (s,d) \in \mathcal{S} \tag{4.5}$$

$$y_{sv}^d \in \{0,1\} \qquad \forall (s,d) \in \mathcal{S}, \forall v \in N_s \tag{4.6}$$

$$\pi_v^s - \pi_s^s + \pi_s^d - \pi_v^d + z_{sv}^d \le 0, \quad 0 \le z_{sv}^d \le 1 \qquad \forall (s,d) \in \mathcal{S}, \forall v \in N_s \tag{4.7}$$

$$\sum_{v\in N_s} z_{sv}^d \ge \alpha_s^d, \quad 0 \le \alpha_s^d \le 1 \qquad \forall (s,d) \in \mathcal{S} \tag{4.8}$$

$$c_{ij} = c_{ji}, \quad c_{ij} \in \{1, \ldots, C_{\max}\} \qquad \forall (i,j) \in E \tag{4.9}$$

In the ILP, (4.4)–(4.6) enforce the Shortest Path Optimality Criteria. In particular, for each destination d and each node s, the node potential π_s^d is set so that the potential-difference $\pi_d^d - \pi_s^d$ encodes the shortest path distance $\text{dist}(s,d)$ from s to d. For each edge (i,j), the constraint $\pi_j^d \le \pi_i^d + c_{ij}$ relates the node potentials to the actual cost setting c, and the binary variable y_{sv}^d is used to guarantee that the inequality is satisfied with strict equality for at least one neighbor v of each s [126]. Namely, y_{sv}^d takes the value 0 if and only if v is a shortest path next-hop from s to d and 1 otherwise, and (4.5) guarantees that for at least one neighbor v of s variable y_{sv}^d will be set to 0.

Furthermore, (4.7)–(4.8) represent the link-protecting LFA condition as of (4.1). In particular, writing (4.7) in a more verbose form we get:

$$(\pi_d^d - \pi_v^d) + z_{sv}^d \le (\pi_s^s - \pi_v^s) + (\pi_d^d - \pi_s^d) \ ,$$

which basically coincides with (4.1) when $z_{sv}^d > 0$ by substituting $\pi_j^j - \pi_i^j = \text{dist}(i,j)$. Correspondingly, z_{sv}^d in fact serves as an indicator variable whose value is positive if and only if v is a link-protecting LFA from s to d. Moreover, (4.8) ensures that

α_s^d only becomes positive if at least one neighbor of s provides LFA towards d. The requirements (4.9) guarantee that costs are symmetric and are selected from the interval $\{1, \ldots, C_{\max}\}$. Finally, the objective function (4.3) maximizes the number of LFA protected node pairs.

There are two problem parameters to the ILP: $C_{\max}$ is the maximum permitted cost, while $C \geq nC_{\max}$ is the maximum allowed potential difference between two neighboring nodes. Then, solving the ILP to optimality yields the link cost setting c that maximizes $\eta^{\mathrm{LP}}(G, c)$ over the input topology G. This can be done by any standard branch-and-bound ILP solver, at least as long as the size of the network is not particularly large (see later). Tightening the ILP, like strengthening the formulation by cutting planes [127], is beyond the scope of this paper.

The ILP is easy to modify to handle the node-protecting version of the LFA cost optimization problem. For this, we need to augment (4.7)–(4.8) with the following constraints:

$$\pi_v^e - \pi_e^e + \pi_e^d - \pi_v^d + z_{sv}^d \leq C y_{se}^d \qquad \forall (s,d) \in \mathcal{S}, \forall v \in N_s, \forall e \in N_s \setminus \{d, v\} \quad (4.10)$$

Here, the new constraints (4.10) will only let z_{sv}^d to take a positive value, indicating that v is a node-protecting $s - d$ LFA, if (4.2) holds in addition to (4.1).

4.3.4 Approximate Algorithms

The above ILP has $O(n^3)$ integer variables, which makes it intractable in anything but the smallest topologies. Therefore, below we provide a set of approximation algorithms, facilitating to obtain a reasonable link cost setting in larger networks as well. In fact, we present a complete family of heuristics, each member of the family having distinct efficiency, running-time, and memory-requirements. This facilitates for picking the best heuristics for the particular requirements.

We chose the simulated annealing probabilistic metaheuristic as the main framework to fund our approximation algorithms onto [128], and within this framework we obtained different heuristics by fine-tuning certain aspects and parameters of the framework. The basic version of the simulated annealing metaheuristic operates as

follows: starting from a randomly chosen initial cost c, choose randomly a neighbor c' "nearby" c. Here, two cost settings are nearby if they differ at exactly one link by exactly 1. If the new cost c' provides larger LFA coverage, then it is unconditionally accepted. On the other hand, if c' is worse then it is still accepted with a certain probability, depending on a system parameter called the temperature. The temperature is set so that from an initial, relatively high value it gradually decreases as the algorithm proceeds, ensuring that the system easily escapes from local optima in the beginning, while it will increasingly tend to get stuck in a good quality optimum eventually. The iteration terminates if the temperature reaches a certain threshold or the algorithm could not improve the LFA coverage after a certain number of steps.

The pseudo-code for the approximation framework is given in Alg. 1. Note that the pseudo-code works the same for the link-protecting and the node-protecting case, therefore we below give a generic treatment suitable to handle both cases. The input to the heuristic is the graph $G(V, E)$, initial temperature T_0 and maximum allowed cost $C_{\max}$, and the output is the final cost c.

Algorithm 1 Heuristic LFA Cost Optimization framework.

```
c ← random_cost(C_max), T ← T_0
while T > 0 and η(G, c) < 1
    c' ← choose_cost(c)
    Δη ← η(G, c) − η(G, c')
    if accept_cost(Δη, T) then
        c ← c'
    end if
    T ← T − 1
end while
```

This framework uses a couple of procedures, yet to be specified.

- random_cost(C): this procedure returns a random initial cost in the range $\{1, \ldots, C_{\max}\}$ for each link. Throughout our numerical studies, we used uniformly distributed initial costs.

- choose_cost(c): this procedure selects a cost "nearby" c. Let neigh(c) denote the set of costs that differ from c at one link by 1. We used two different cost selection policies: we either chose a nearby cost randomly (choose_random_cost(c)),

or we chose greedily the cost setting that improved LFA coverage the most (choose_greedy_cost(c)), i.e, sets c' according to $\text{argmax}_{q \in \text{neigh}(c)} \eta(G, q)$.

- accept_cost($\Delta\eta, T$): this procedure guides the way the new cost setting c' is accepted. Again, we used two different policies. Both policies share the property that a cost that improves LFA coverage is accepted unconditionally. In addition, the policy proportional_test($\Delta\eta, T$) accepts the new cost even if worse with probability proportional to the temperature. Correspondingly, proportional_test($\Delta\eta, T$) returns true if $\Delta\eta < 0$ or $T >$ random(T_0). Here, the procedure random(x) returns a uniformly distributed random sample from $[0, x]$. The so called *Metropolis-test* (metropolis_test($\Delta\eta, T$)), on the other hand, returns true if $\Delta\eta < 0$ or random(1) $< \exp(-\Delta\eta/T)$.

A useful consequence of defining our approximation framework in a general form mentioned above is that different choices for the input parameters as well as the selection of the procedures choose_cost(c) and accept_cost($\Delta\eta, T$) yield different heuristics, with drastically varying performance and running time. For instance, setting the initial temperature low causes faster termination but reduces the probability of finding the global optimum, since this allows the algorithm to explore only a limited domain of the problem space. Furthermore, picking the choose_greedy_cost(c) procedure for selecting the best candidate nearby cost has the potential to rapidly improve the costs initially, but might cause overly long running time due to having to check LFA coverage for each neighboring cost in each iteration. The choice for the accept_cost($\Delta\eta, T$) procedure, on the other hand, influences the proneness of the heuristics to get stuck in local optima.

We also experimented with several small modifications of the above basic approximation framework, in an attempt to obtain good solutions [128].

- *Tabu lists* are useful to preclude the iteration from oscillating between two ore more cost settings, through prohibiting the algorithm to revisit a certain number of previously visited solutions.

- *Restarting* allows the iteration to be restarted from an earlier good solution in the case when the algorithm gets stuck in a local optimum. Note that no temperature reduction is made during the restart.

- *Quantum tunneling* is similar to restarting, but instead of jumping back to a previous solution we rather jump to another random solution. Again, the intention is to avoid sticking in local optima. In our implementation, if after a configurable number of iterations LFA coverage could not be improved then the heuristics set each link cost randomly up or down by the average link cost. This will change at least one shortest path in the network, and so the iteration transitions into a different domain of the problem space.

The running time of the heuristics principally depends on the choice of the choose_cost(c) procedure. With greedy selection, the complexity is $O(T_0 mn^3)$, dominated by the need of evaluating $\eta(G, q)$ (needing $O(n^3)$ steps) in each iteration for each $2m$ neighbor q of the current cost c. With selecting the choose_random_cost(c) procedure, on the other hand, complexity decreases to $O(T_0 n^3)$, as LFA coverage needs to be evaluated in each iteration only once. This, however, comes with a substantial drop in efficiency, as evidenced by the numerical results presented next.

Finally, we call the attention to an appealing aspect of our heuristic framework. In particular, the framework is easy to adapt for different operational requirements not explicitly addressed in the dissertation. For instance, the heuristics are completely neutral to whether the underlying graph representation is undirected or directed (a case more relevant to IP networks), or whether link costs are symmetric or asymmetric. The framework does not even require link costs to be integral. Further operational issues, like traffic engineering concerns, suppressing equal cost shortest paths [124], or considering broadcast LANs, etc., are also easy to incorporate into the optimization algorithms through tuning the objective function. The same applies to more elaborate failure models, like multiple failures, Shared Risk Link Groups, etc. Discovering the breadth of these options is beyond the scope of this Dissertation.

4.3.5 Numerical Evaluations

In the course of our numerical studies, first we were curious as to how close the approximate LFA Cost Optimization algorithms can get to the optimum. Therefore, we implemented the ILP (4.4)–(4.9) and the approximation framework described in Section 4.3.4. Below, only results for the greedy cost selection rule (choose_greedy_cost(c)) and the temperature-proportional acceptance rule (proportional_test($\Delta\eta, T$)) are given,

with a tabu list of size 20, no restarting and no quantum tunneling, as this proved most efficient in comparison studies to be discussed later. The simulated annealing procedure was executed 500 times consecutively ($T_0 = 150$, $C_{\max} = 20$) and the cost c^* that attained the highest LFA coverage was selected. First, we only give the results for the link-protecting case.

We found that about the largest non-trivial graphs for which the ILP can be solved to optimality are of 8 nodes. Unfortunately, very few real topologies of this small size are available in the literature. Thus, the first round of the evaluations were run on Erdős-Rényi random graphs ($n = 8$, expected node degree 3). Out of the 20 random graphs generated, 17 was 2-connected, and results are only given for these instances. Table 4.1 gives some characteristics of the graphs (number of nodes n, and number of links m); the theoretical lower and upper bounds on LFA coverage (with respect to Proposition 3 and Proposition 4); and the actual LFA coverage $\eta(G, c_{\text{opt}})$ for the costs c_{opt} obtained by the ILP and the above customized greedy version of the approximation algorithm (η^*_{gr}). We observe that from the 17 experiments only in 2 cases the approximation did not find the optimum (these experiments are marked by an asterisk in Table 4.1), and the difference is at most $2 - 3\%$ in LFA coverage. This indicates that in small networks the simulated-annealing-based heuristics perform quite efficiently. Additionally, it can be observed that the theoretical bounds provide a solid estimate on the LFA coverage.

In the second round, we examined the performance of the different approximate LFA Cost Optimization algorithms we proposed in the previous section in larger real network topologies mentioned in Section 2.3, where the ILP could not be solved to optimality. Additionally, we also ran the evaluations on some artificial topologies with uniform costs. In particular, M_n are the Möbius ladder graphs of n nodes and $n + \frac{n}{2}$ links as discussed in Section 4.3. First, we deal with the link-protecting case, while the case of node protection will be discussed subsequently.

The particular approximation algorithms we used were as follows. First, we chose the "textbook" version of the simulated annealing algorithm, with random cost selection (choose_random_cost(c)) and the Metropolis-test for cost acceptance (metropolis_test($\Delta\eta, T$)), with a tabu list of size 20 and different customized settings for the restarting and quantum tunneling thresholds. The notation η^* will be used to denote the LFA coverage as provided by this textbook version of the heuristics. In

Table 4.1. LFA Cost Optimization in random topologies

Num	n	m	Lower/Upper	$\eta(G, c_{\text{opt}})$	η^*_{gr}
1*	7	11	0.278/1	1	0.976
2	8	9	0.095/0.571	0.536	0.536
3*	8	13	0.214/1	1	0.982
4	7	11	0.278/1	1	1
6	8	9	0.143/0.571	0.571	0.571
9	7	11	0.208/1	0.952	0.952
10	8	11	0.114/1	0.857	0.857
11	8	10	0.143/0.857	0.75	0.75
12	8	9	0.095/0.571	0.429	0.429
13	8	11	0.143/1	0.911	0.911
14	8	11	0.19/1	0.821	0.821
15	8	11	0.19/1	0.946	0.946
16	7	8	0.111/0.667	0.5	0.5
17	8	14	0.2/1	1	1
18	8	11	0.114/1	0.714	0.714
19	8	9	0.143/0.571	0.482	0.482
20	8	10	0.143/0.857	0.679	0.679

particular, $\eta^*_{Q=q}$ will signify the version with the quantum tunneling threshold set to q (i.e., after q unsuccessful trials the algorithm jumps to a new cost set). We used the setting $q = 1$, $q = 10$ and $q = 20$. In addition, $\eta^*_{R=r}$ denotes the LFA coverage by setting the restarting threshold in a similar vein. Evaluations were run for $r = 2$, $r = 10$, and $r = 20$. Finally, we also executed the greedy version as used in the first simulation round (choose_greedy_cost(c), proportional_test($\Delta\eta, T$), tabu list of size 20, no restarting and no quantum tunneling). Results are again marked by η^*_{gr}. For each topology, the algorithms were executed 1000 times ($T_0 = 1000$, $C_{\max} = 20$) and the best cost c^* was selected. There was only one topology on which we could solve the ILP to optimality: AS1221. For this particular network, each of the approximation algorithms could attain the ILP optimum ($\eta(G, c_{\text{opt}}) = 0.833$).

Detailed results of link-protecting mode are presented in Table 4.2. The columns mean (in the order of appearance): the characteristics of the topologies (name and the average node degree Δ); the LFA coverage obtained by the original link cost setting for the graphs $\eta(G, c)$; and the LFA coverage obtained by the different approximation algorithms. We also highlight the results for some select topologies in Fig. 4.6.

Our observations are as follows. First, we found that the LFA coverage produced by the approximate algorithms is usually significantly higher (about 90% or more in the link-protecting case) than the LFA coverage produced by the network's original

Table 4.2. Link-protecting results for the LFA Cost Optimization heuristics in real and artificial topologies

Name	Δ	$\eta(G,c)$	η^*	$\eta^*_{Q=1}$	$\eta^*_{Q=10}$	$\eta^*_{Q=20}$	$\eta^*_{R=2}$	$\eta^*_{R=10}$	$\eta^*_{R=20}$	η^*_{gr}
AS1221	2.57	0.809	0.833	0.833	0.833	0.833	0.833	0.833	0.833	0.833
AS1239	4.60	0.873	0.958	0.958	0.959	0.959	0.96	0.959	0.958	0.963
AS1755	3.66	0.872	0.983	0.983	0.983	0.983	0.983	0.98	0.98	0.993
AS3257	4.74	0.923	0.997	0.995	0.998	0.997	0.997	0.997	0.997	1
AS3967	3.42	0.785	0.971	0.973	0.973	0.966	0.976	0.966	0.971	0.983
AS6461	4.35	0.933	1	0.996	0.996	0.996	1	1	1	1
Abilene	2.5	0.56	0.674	0.674	0.674	0.674	0.674	0.674	0.674	0.674
Arnes	2.78	0.623	0.702	0.704	0.707	0.704	0.706	0.703	0.7	0.709
AT&T	3.45	0.822	0.982	0.984	0.982	0.98	0.98	0.978	0.984	0.987
Deltacom	2.85	0.577	0.654	0.658	0.652	0.659	0.652	0.661	0.651	0.662
Geant	2.97	0.69	0.74	0.742	0.745	0.743	0.74	0.741	0.737	0.76
Germ_50	3.52	0.9	0.929	0.931	0.935	0.93	0.932	0.93	0.939	0.966
Germany	2.94	0.695	0.9	0.893	0.893	0.904	0.904	0.904	0.9	0.911
InternetMCI	3.47	0.904	0.932	0.932	0.932	0.932	0.932	0.932	0.932	0.932
Italy	3.39	0.784	0.926	0.928	0.922	0.932	0.927	0.93	0.922	0.944
NSF	3.3	0.86	0.949	0.955	0.964	0.95	0.958	0.956	0.955	0.977
M_6	3	0.4	1	1	1	1	1	1	1	1
M_{10}	3	0.444	0.922	0.922	0.933	0.933	0.922	0.933	0.922	1
M_{18}	3	0.470	0.882	0.875	0.885	0.885	0.882	0.872	0.888	0.905
M_{30}	3	0.482	0.889	0.883	0.886	0.886	0.889	0.889	0.888	0.904

cost setting (around 70% on average). The improvement almost always exceeds 7%, but in many cases it amounts to more than 20-25% (e.g., AS3967, or the German backbone). This suggests that optimizing costs specifically for LFA usually attains significant improvement in network resilience. The improvement is especially significant for the artificial networks. We also found that the denser the network, the higher the LFA coverage. It seems that networks with an average node degree exceeding about 3.5 lend themselves especially well to LFA Cost Optimization (AS1239, AS1755, AS3257, AS6461, AT&T, Germ_50): in these networks even the default cost settings yield a higher than 80% LFA coverage and our cost optimization tool can bring these networks well beyond 95% and close to 100% in many cases. Networks of average node degree of 3 are still amenable to LFA, but when the degree falls below 3 the chances of getting a high LFA coverage rapidly vanish. For sparser networks (like the Abilene topology), the final LFA coverage $\eta(G,c^*)$ hardly reached 70%. These observations are in line with the propositions mentioned in Section 4.2. Note, however, that node degree alone is not sufficient to assess the extent to which LFA can protect a network, as there are topologies (the Möbius ladder graphs) that have small

Figure 4.6. Final link-protecting LFA coverages for some select topologies attained by the different heuristics

average degree of 3 but complete link-protecting LFA coverage over some appropriately chosen costs. It seems that LFA Cost Optimization is most difficult when the degree is about 3.

Second, we observe that for large Möbius ladder graphs the approximation could not get closer than 10% to the optimum (which we know is $\eta(G, c_{\text{opt}}) = 1$ in this case). This indicates that in larger topologies the efficiency of the heuristics we identified in small networks might not be present.

Third, we find that the approximation algorithms work roughly similarly. In particular, Fig. 4.7 compares the LFA coverage of the different heuristics, when taking the textbook simulated annealing heuristics as the basis for the comparison. We observe that the best performance is attained by the greedy version consistently. This is not surprising, considering the more elaborate (and more time-consuming) cost selection rule. In addition, quantum tunneling seems to work better for larger thresholds, while restarting produces consistently better results for smaller thresholds. The differences in the eventual LFA coverage between the different heuristics, however, are in the order of mere percents. We note, however, that minuscule differences in the LFA coverage can mean dozens of more protected source-destinations in reality.

Figure 4.7. Average final LFA coverages obtained by different heuristics compared to the textbook version

This is because the normalizing factor $n(n-1)$ in (2.1) can become very large in big networks. For instance, in the Deltacom topology the greedy heuristics covered 2367 source-destination pairs while the others reached less than 2300.

Last but not least, the execution time for 1000 rounds of the different heuristics is depicted in Fig. 4.8 in a logarithmic scale. As expected, for the wins in performance with the greedy version of the heuristics we pay by significantly increasing execution time. This is because, as mentioned previously, choosing the best neighbor in every step requires $2m$ times more evaluations of the LFA coverage metric than selecting one randomly. Note that these execution times are representative to the offline phase of LFA Cost Optimization, which is usually performed only once for the lifetime of the network before the final deployment, and in no way affect the *real-time* requirements of finding loop-free alternates in the online phase (which remains below 50ms as required). Therefore, we do not consider execution times to be a particularly pressing issue in LFA Cost Optimization. However, it should be noted that during the network's lifetime in the course of manual interventions caused by network operators a topology could change for longer period or even for ever. In such cases, execution time of the algorithms could play an important role. In Fig. 4.9, the average execution

Figure 4.8. Execution times for different heuristics in hours

times for the different heuristics are depicted from left to right in a descending order of the LFA coverage they could attain. One can observe that for the best results (attained by η^*_{gr}) we have to wait the most. Furthermore, on average the heuristics that use quantum tunneling with thresholds 20 and 10 seem to be feasible solutions if higher coverage is required but in much shorter time period than the execution time needed by the greedy version.

Finally, we mention that, curiously, running 1000 rounds of the heuristics is usually unnecessary, because the best solutions were realized after 200 rounds in each case.

We repeated the simulations for the node-protecting case as well (see Table 4.3). The observations are essentially the same as in the link-protecting case, just the numeric values seem somewhat smaller. We found that the initial node-protecting LFA coverage was 57% on average which the algorithms could improve by about $10-20\%$, so that eventual the LFA coverage was in the range of $75-80\%$. In some cases, however, the heuristics could reach more than 25% of improvement (for instance, this is the case for the AS3257 topology). Again, the greedy heuristic is consistently the most effective, and restarting and quantum tunneling seem worthwhile extensions to textbook simulated annealing.

Figure 4.9. Average execution times for different heuristics in logarithmic scale. The heuristics are in descending order from left to right according to their attained LFA coverage

Table 4.3. Node-protecting results for the LFA Cost Optimization heuristics in real and artificial topologies

Name	Δ	$\eta(G,c)$	η^*	$\eta^*_{Q=1}$	$\eta^*_{Q=10}$	$\eta^*_{Q=20}$	$\eta^*_{R=2}$	$\eta^*_{R=10}$	$\eta^*_{R=20}$	η^*_{gr}
AS1221	2.57	0.452	0.523	0.523	0.523	0.523	0.523	0.523	0.523	0.523
AS1239	4.60	0.757	0.886	0.896	0.889	0.886	0.894	0.893	0.882	0.937
AS1755	3.66	0.764	0.895	0.885	0.888	0.885	0.882	0.882	0.911	0.941
AS3257	4.74	0.726	0.863	0.877	0.881	0.871	0.871	0.873	0.87	0.938
AS3967	3.42	0.642	0.84	0.838	0.838	0.847	0.847	0.857	0.835	0.897
AS6461	4.35	0.738	0.845	0.838	0.83	0.841	0.852	0.841	0.838	0.886
Abilene	2.5	0.515	0.606	0.598	0.606	0.606	0.606	0.606	0.606	0.606
Arnes	2.78	0.359	0.469	0.482	0.484	0.478	0.471	0.472	0.486	0.49
AT&T	3.45	0.58	0.783	0.8	0.796	0.783	0.779	0.781	0.781	0.82
Deltacom	2.85	0.488	0.559	0.571	0.566	0.568	0.567	0.564	0.558	0.581
Geant	2.97	0.41	0.572	0.590	0.585	0.581	0.577	0.573	0.57	0.622
Germ_50	3.52	0.827	0.829	0.816	0.807	0.815	0.829	0.822	0.82	0.86
Germany	2.94	0.562	0.731	0.72	0.709	0.713	0.713	0.705	0.705	0.727
InternetMCI	3.47	0.704	0.783	0.766	0.766	0.777	0.78	0.777	0.769	0.809
Italy	3.39	0.57	0.768	0.754	0.748	0.75	0.764	0.754	0.754	0.803
NSF	3.3	0.633	0.83	0.796	0.806	0.815	0.829	0.807	0.836	0.866
M_6	3	0.444	0.922	0.888	0.9	0.9	0.9	0.911	0.9	0.966
M_{10}	3	0.47	0.81	0.8	0.807	0.81	0.816	0.813	0.81	0.849
M_{18}	3	0.482	0.812	0.809	0.809	0.81	0.809	0.817	0.812	0.833
M_{30}	3	0.4	1	1	1	1	1	1	1	1

In summary, our results suggest that most real network topologies, which are usually richly connected and highly redundant, lend themselves readily to LFA Cost Optimization, to the point that almost perfect link-protecting LFA coverage can be achieved in many cases. For fast execution time, the textbook simulated annealing version with a reasonable quantum tunneling and restarting threshold seems most appealing, whereas the greedy version is the best choice when the aim is to find the highest quality link costs.

4.4 Combined LFA Network Optimization

So far, LFA Graph Extension problem and the aforementioned LFA Cost Optimization problem have been studied separately. However, in certain cases a network operator may not be able to rely solely one of them. The cost of augmenting the network with new physical links can be reasonable expensive, and in certain cases only one additional link would be required to attain the desired reliability. On the other hand, when other objectives, such as traffic engineering or load balancing play a more important role, altering the link costs and by this way affecting certain shortest paths may cannot be tolerated. Moreover, as it was discussed in Section 1.6, an additional link may alters existing shortest paths, which can be compensated by optimizing link costs at the same time. In these cases, an efficient combination of the aforementioned two methods could help network operators to find a compromise solution.

Hence, in this section we propose the *Combined LFA Network Optimization problem*, which augment the network with new links and optimize the link costs at the same time to achieve high LFA failure case coverage. This problem can be formulated as follows:

Definition 6. *LFACombinedOptLP(G, $\mathcal{S}$, k): Given a simple, undirected, weighted graph $G(V, E)$, a set of source-destination pairs $\mathcal{S}$, and a positive integer k, is there a set $F \subseteq \overline{E}$ with $|F| \leq k$ and properly chosen cost function c, so that $\eta_{\mathcal{S}}^{LP}(G(V, E \cup F), c) = 1$?*

The difference from the LFA Graph Extension problem is that in the above formulation, we allow for link costs and, consequently, the shortest paths to change. Note that for $k = 0$ we obtain the LFA Cost Optimization problem, which immediately

implies the following result.

Theorem 4. *The Combined LFA Network Optimization problem is NP-complete.*

From all the optimization problems treated so far, Combined LFA Network Optimization is the most difficult, since both sub problems it covers are NP-complete. Therefore, instead of aiming for an optimal solution, we propose a heuristic algorithm based on the consecutive application of the aforementioned heuristics presented for the individual sub problems. In particular, in every iteration we executed an LFA Graph Extension phase followed by an LFA Cost Optimization phase. In the LFA Graph Extension phase, we add l new links to the network (where l is a problem parameter) by using the LJC algorithm [103]. This algorithm is chosen because it promises the largest increase in LFA coverage. In the LFA Cost Optimization phase, we compute a link cost setting that approximately maximizes the LFA coverage on the augmented graph obtained in the previous phase. Since our aim is to improve the LFA coverage as much as possible, for the LFA Cost Optimization phase we used the best performing greedy version from our heuristics (denoted by η^*_{gr} in Section 4.3.5).

The two phases are applied iteratively one after the other, until the LFA coverage reaches 1. Note that LFA Graph Extension always guarantee 100% LFA failure case coverage, since in the worst case, it will result a complete graph, which is fully LFA-protected against single failures.

4.4.1 Numerical Evaluations

In order to evaluate the performance of the combined algorithm compared to the individual algorithms, we conducted several rounds of numerical experiments. Herein, we present the results for only a subset of the network topologies. Similar results were obtained for the rest of the topologies as well. Furthermore, only the case of link protection are presented. In case of node protection, a similar pace can be realized.

First, we were curious as to what is the optimal setting for the parameter l. We experimented with the settings $l \in [1, 2, 3]$ as in [84] it was found that at most 3 new links are always enough to realize a reasonable improvement in LFA coverage.

The results obtained on some selected topologies can be seen in Fig. 4.10. For each network topology, the *red line with circles* (LFA GE(1)) represents the LFA coverage realized by using LFA Graph Extension exclusively (i.e., with no LFA Cost

(a) AS1755 (b) Abilene

(c) Germany (d) Italy

Figure 4.10. Results of the several combined metrics run on some selected topologies

Optimization phase applied), and the *green line with triangles* (LFA GE(1) + CO), the *blue line with crosses* (LFA GE(2) + CO) and the *purple line with rectangles* (LFA GE(3) + CO) give the LFA coverages attained by the combined algorithm for the cases of $l = 1$, $l = 2$, and $l = 3$, respectively. Easily, for the case of $l = 1$, we have a data point for each iteration, but for other cases some of these internal data points are missing, as improvement only appears after adding more than one link in such cases. The results show that the combined algorithm performs best when in every LFA Graph Extension phase we add only a single link, and this is immediately followed by a cost optimization phase. In consequence, we used the setting $l = 1$ in the rest of the study.

After fixing l, we turn to the main question of our numerical studies: to what extent the combined algorithm outperforms previous algorithms? The results for some selected topologies are given in Table 4.4, which shows in order of the appearance:

Table 4.4. Results for the LFA Graph Extension and the Combined LFA Network Optimization algorithms

Topology		LFA Graph Extension							Combined LFA Network Optimization						
Name	$\eta(G,c)$	1	2	3	4	5	6	#	1	2	3	4	5	6	#
AS1239	0.873	0.932	0.971	0.985	0.992	0.997	1	**6**	0.994	0.998	1	–	–	–	**3**
AS1755	0.872	0.947	0.964	0.973	0.980	0.986	0.993	**8**	0.986	0.993	1	–	–	–	**3**
AS3257	0.923	0.971	0.983	0.987	0.990	0.991	0.993	**11**	0.998	1	–	–	–	–	**2**
AS6461	0.933	0.985	0.996	1	–	–	–	**3**	1	–	–	–	–	–	**1**
Abilene	0.56	0.757	0.833	0.894	0.947	0.992	1	**6**	0.871	0.939	0.962*	0.977*	0.992	1	**6**
Italy	0.784	0.830	0.862	0.891	0.911	0.928	0.943	**20**	0.970	0.980*	0.985*	0.991*	0.994*	0.995	**11**
Germany	0.695	0.786	0.841	0.882	0.920	0.948	0.966	**12**	0.937	0.959*	0.966*	0.974*	0.981*	0.988*	**9**
InternetMCI	0.904	0.973	0.985	0.994	0.997	1	–	**5**	0.991	0.994*	1	–	–	–	**4**

name of the topology; the initial LFA coverage $\eta(G,c)$; LFA coverage achieved with simple LFA Graph Extension in step $1,2,3,4,5,6$, and the number of links needed for full coverage (denoted by hashmark); and finally the same results obtained by the Combined LFA Network Optimization algorithm. For the combined algorithm, the LFA coverage values marked with asterisks highlight the cases when the LFA Cost Optimization phase did not improve LFA coverage at all (for instance, in case of Abilene network, after adding 3 new links, the LFA Cost Optimization phase could not improve the LFA coverage of 0.962, so it was attained merely by the LFA Graph Extension phase). Our results suggest that the combined algorithm reaches complete coverage with much fewer links than the pure LFA Graph Extension algorithm (the difference is highlighted in Fig. 4.11).

This is not surprising, as the combined algorithm is free to reconfigure link costs, which the LFA Graph Extension algorithm is not permitted to do. We found that the combined algorithm significantly reduces (on average by more than 50%) the number of additional links necessary for reaching 100% LFA coverage. Similarly to the performance of using LFA Cost Optimization exclusively, we also found that it is not worth running all the 500 rounds of the LFA Cost Optimization algorithm, because in the combined algorithm, the optimum was always realized in less than 200 rounds.

Figure 4.11. Number of links needed to be added using the different algorithms

4.5 Summary

In this chapter, we proposed two network optimization algorithms for improving the level of fast IP-level resilience using Loop-Free Alternates. In particular, we presented LFA Cost Optimization problem, which, instead of adding new links, rather calls for modifying link costs to instantiate new LFAs. We showed that this problem is NP-complete and we gave an Integer Linear Program to obtain an exact solution. As our exact algorithm only worked in small networks, we proposed a family of simulated-annealing-based approximations with different tunable parameters in order to increase the performance and/or decrease the execution time. Our heuristics could achieve significant boost in LFA coverage in many real-world network topologies, to the point that in some cases close to perfect protection could be realized. However, we also found that some topologies are less amenable to LFA Cost Optimization, therefore we defined a combined formulation of LFA Graph Extension and LFA Cost Optimization, called LFA Combined Optimization, and we studied to what extent this combination can be effective for LFA-based network optimization. Among all optimization approaches, we found that the combined algorithm produces the best results, indicating that in many real networks, complete LFA-based protection is

attainable with adding fewer number of new links than it was needed by the LFA Graph Extension. Considering that LFA is just a router-configuration command away in many modern IP networks, we believe that the wide spectrum of LFA network optimization strategies presented in this chapter provides a rich set of options for operators to choose from, according to their own preference whether it is economically more feasible to add new links, change costs, or do both, to improve LFA-protection in their network.

Chapter 5

Analysis of Remote LFA Failure Case Coverage

Up to not so long ago, Loop-Free Alternates (LFA) was the only viable option for providing fast protection in pure IP and MPLS/LDP networks. Unfortunately, as it was shown in the previous chapter, LFA cannot provide protection for all possible failure cases in general. Recently, the IETF has initiated the Remote Loop-Free Alternates (rLFA) technique as a simple extension to LFA, to boost the fraction of failure cases covered by fast protection. Before further standardization and deployment, however, it is crucial to determine to what extent rLFA can improve the level of protection against single link or node failures in a general IP network.

In this chapter, we take the first steps towards this direction. As a first approach, we limit our attention to the special case when link costs are uniform, since as shall be shown, results for LFA can only be generalized to rLFA under the unit cost assumption. Anyway, as it was discussed in Section 3.4, considering unweighted networks is important for several further reasons as well.

This chapter is essentially a crystallization and an extension of the rLFA specification [80]. Beyond dealing with link failures, our analysis also covers the crucial case of single node failures[1]. We provide basic graph theoretical toolset for analyzing rLFA failure case coverage in the case when link costs are uniform, and we establish a sufficient and necessary condition for a network to have 100% rLFA failure coverage. We also study the "bad cases" for rLFA, in which failure coverage is particularly

[1]Recently, the IETF has initiated an Internet Draft for covering this case [129].

poor. Building on [80], we distinguish between *plain* and *extended Remote LFA* and we quantify the benefits that come from the usage of extended rLFA.

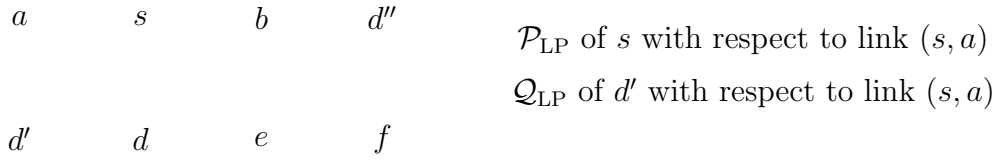
5.1 Remote LFA in Detail

As mentioned in Chapter 4, in Loop-Free Alternates, the backup routes are precomputed and installed in the router as backup for the primary routes. Once a router detects a link or adjacent node failure, it switches to a backup route to avoid traffic loss. While LFA considers only physically adjacent routers for computing backup routes, Remote LFA enables a backup next-hop to be more than one hop away. After a failure, an adjacent router recognizes it and tries to find a (remote) node whose shortest path to the destination is unaffected by the failed component. If such a router is found, then packets will be forwarded to it. Remote LFA relies on tunnels to provide additional logical links towards backup next-hops. After the remote node receives the packets, it forwards them to the primary destination. Note that the tunneled traffic is restricted to shortest paths just like "normal" traffic, hence the tunnel must avoid the failure as well.

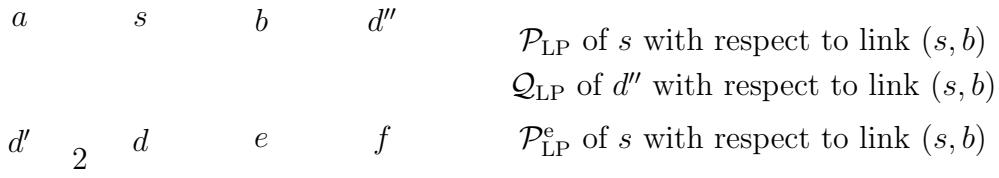
Below, we discuss the conditions and different protection schemes of Remote LFA formally. Consider the network depicted in Fig. 5.1(a). Suppose that router s wishes to send a packet to destination d'. As it was already shown in Section 1.5, if the link (s, a) fails, s has no neighbor that would not pass the packet back if it were chosen as a bypass, so in this case the given source-destination pair cannot be protected via standard LFA. However, if a tunnel is created between s and e (marked by black dashed line), then e, now being a direct neighbor of s, would become an LFA for d', thereby protecting the link (s, a).

There are numerous tunneling mechanisms, which fulfill the requirements of this design. In an MPLS/LDP enabled network, for instance, simple label stack (discussed in Chapter 1.1.1) can be used to provide the required tunnel without any additional modification to the IP header of the packets.

Next, consider Fig. 5.1(b) where node s remains the source but node d'' becomes the destination and the link (s, b) fails. Then, (s, b) cannot be protected for a lack of a suitable tunnel, since all nodes whose shortest path does not go through (s, b) can only be reached from s through (s, b) itself. This suggests that while the use of rLFA



(a) Higher protection can be attained with plain rLFA



(b) A basic situation that cannot be protected with plain rLFA either

Figure 5.1. Sample network topologies with uniform link costs. Solid lines mark the IP network topology, the thick dashed line denotes the tunnel, while thick solid arrows denote the default shortest paths from node s to node d' and to node d'', respectively

definitely can provide higher protection level against link failures than pure LFA, it still does not facilitate full protection for all failure cases in a general topology.

Next, we discuss how all the previously mentioned properties change when node protection is also taken into account. Consider the network depicted in Fig. 5.2. Again, suppose that node s wants to send a packet to destination node d. The next-hop of s to d is node e. One can easily check that if link (s, e) goes down then node n and m are suitable repair tunnel endpoints, since their shortest paths to node d avoid the failed component[2] and they can also be reached without traversing link (s, e). However, if not only the link (s, e) fails but the node e itself, then node m can be the one and only remote loop-free alternate, since node n has ECMP to node d through the failed node e. Similarly to the case of node-protecting LFA, it should be also noted that in case of node protection, we have to deal with the *last-hop problem*, which again refers to the situation when the destination node itself goes down that obviously cannot be protected. As discussed above, this can be handled differently. During our Remote LFA analysis, node protection between two neighboring nodes is considered as *undefined* instead of falling back to link-protecting possibilities. However, when assuming the latter would result in different behavior we shall examine both cases.

[2]Shortest path from node m to node d is $m \to d$, while node n has two shortest paths to d, namely $n \to e \to d$ and $n \to m \to d$.

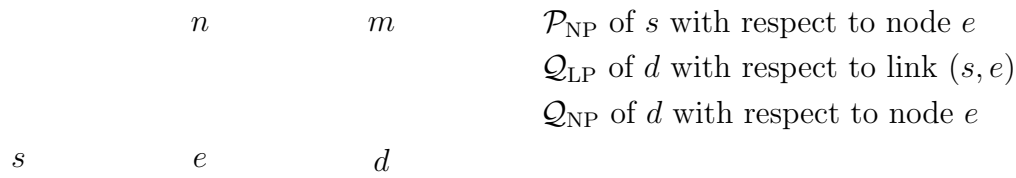


Figure 5.2. A sample network topology for illustrating how node protection differs from link protection

5.1.1 Identifying Remote LFA Staging Point

During a failure, the repair tunnel endpoint needs to be a node in the network reachable from the source without traversing the failed component. In addition, the repair tunnel endpoint needs to be a node from which packets will normally flow towards their destinations without being attracted back to the failed component. Correspondingly, in the case of a link failure, the set of routers which can be reached from a source without traversing the failed link is termed the *P-space* [49] of the source with respect to the failed link (hereafter $\mathcal{P}_{\mathrm{LP}}$, where the subscript LP refers to link-protecting case). Since the source router will only use a repair path when it has detected the failure of the link, the initial hop of the repair path needs *not* be subject to the source's normal forwarding decision process. Therefore, the term *extended P-space* (hereafter $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$) was also defined, which is the union of the $\mathcal{P}_{\mathrm{LP}}$s of each of the source's neighbors. The usage of $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ may enable the source router to reach potential repair tunnel endpoints that were otherwise unreachable. Furthermore, the set of routers from which the destination can be reached without traversing the failed link is termed the *Q-space* (hereafter $\mathcal{Q}_{\mathrm{LP}}$) of the destination with respect to the failed link. The intersection of the source's $\mathcal{P}_{\mathrm{LP}}$ and the destination's $\mathcal{Q}_{\mathrm{LP}}$ with respect to the failed link defines the viable repair tunnel endpoints, known as $\mathrm{PQ}_{\mathrm{LP}}$-nodes, which are practically the Remote LFAs.

As can be seen, for the case of the example network depicted in Fig. 5.1(a) there is only one node (e) that protects the link (s, a), assuming that node s wants to send a packet to node d'. However, considering d'' as the destination $\mathcal{P}_{\mathrm{LP}}$ and $\mathcal{Q}_{\mathrm{LP}}$ turn out different (see Fig. 5.1(b)). Now, there is no intersection of s' $\mathcal{P}_{\mathrm{LP}}$ and $\mathcal{Q}_{\mathrm{LP}}$ of d'', thus viable $\mathrm{PQ}_{\mathrm{LP}}$-nodes do only exist if $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ is used, since if s can pass the packet to

d, then node d will not pass the packet back and the packet transmission will avoid the failed link (s, b).

Next, we extend these definitions to the case of node failures. Note that the rLFA specification [80] does not consider this case, so ours is the first such extension[3]. As it turns out, the case of node protection hardly differs from the case of link protection. When the next-hop fails, the possible repair tunnel endpoint needs to be a (remote) node, which is reached from the source without traversing that failed next-hop itself (instead of only the link to it, as before). Hence, the set of such routers is termed the $\mathcal{P}_{\text{NP}}$ (where the subscript NP refers to node-protecting case) of the source with respect to the failed node. As it was in the case of link protection, the term *extended P-space* (hereafter $\mathcal{P}^{\text{e}}_{\text{NP}}$) can also be defined as the union of $\mathcal{P}_{\text{NP}}$s of each of the source's remaining neighbors. The set of routers whose shortest path to destination bypasses the failed node is termed the $\mathcal{Q}_{\text{NP}}$ of the destination with respect to the failed node. Here again, the intersection of the source's $\mathcal{P}_{\text{NP}}$ and the destination's $\mathcal{Q}_{\text{NP}}$ with respect to the failed node defines the viable repair tunnel endpoints, known as PQ_{NP}-nodes. For the sake of easy comprehension, see Fig. 5.2 and consider node s as source and node d as destination.

In this Dissertation, we slightly diverge from the terminology of the specification [80] and we say that a node is Remote LFA if it is in the intersection of the "simple" $\mathcal{P}_{\text{LP}}$ ($\mathcal{P}_{\text{NP}}$) and $\mathcal{Q}_{\text{LP}}$ ($\mathcal{Q}_{\text{NP}}$, respectively) and we shall use the term *"Extended Remote LFA"* henceforth when $\mathcal{P}^{\text{e}}_{\text{LP}}$ ($\mathcal{P}^{\text{e}}_{\text{NP}}$) is also to be considered for defining the rLFA nodes, i.e., PQ_{LP}-nodes (PQ_{NP}-nodes).

Most of our analysis will be given for "plain" rLFA, as this technique can be easier handled and the problem itself is more tangible. Extended rLFA, on the other hand, requires more attention. Thus, in this Dissertation we mostly treat the "plain" rLFA case, and only highlight some important aspects of "Extended Remote LFAs". Note that, however, from an implementation point of view, there is no difference between them as reaching a remote neighbor in the P-space or in the extended P-space, for instance, requires the same MPLS/LDP label stacking method mentioned in Section 1.1.1[4].

[3]Recently, an Internet Draft has been proposed [129] by the IETF to take into account node-protecting Remote LFA as well.

[4]Note that this was only crystallized in the second version of the Remote LFA draft.

From the above discussion, it is clear that in general not all nodes have LFA or even Remote LFA protection to every other node.

5.2 A Mathematical Toolset for Remote LFA

Below, we give some basic machinery to handle Remote LFAs somewhat more plausibly[5] than what is provided by the mere definitions of P-spaces, Q-spaces, and extended P-spaces specified in [80]. In particular, we define these "spaces" in terms of distance functions that were used in the pure LFA specification [47] as well (recall Definition 2, and Definition 3). As shall be shown, by means of these formal definitions pure LFA and rLFA can be easier compared with each other. Furthermore, we show that, under the unit-cost assumption, all networks are fully protected against single link failures by "Extended Remote LFA". However, this can not be stated for the case of single node failures.

We shall separate the discussion into two parts. First, in line with the specification [80], we consider only single link failures. Then, in the second part, we extend our techniques to single node failures as well.

5.2.1 Link-protecting Case

An arbitrary failed link along the shortest path between a source and a destination can only be protected if the intersection of $\mathcal{P}_{\text{LP}}$ of the source and the $\mathcal{Q}_{\text{LP}}$ of the destination is not empty. First, we show an alternative characterization for $rLFA_{\text{LP}}$ that, as shall be seen, is more amenable to theoretical analysis. Consider the below reformulation of this requirement in terms of the shortest path distance function *dist*.

Definition 7. *For a source node s and next-hop e, some $n \in V$ is in $\mathcal{P}_{\text{LP}}(s, e)$ if and only if*

$$\text{dist}(s, n) < \text{dist}(s, e) + \text{dist}(e, n) \ , \tag{5.1}$$

and some $n \in V$ is in $\mathcal{Q}_{\text{LP}}(s, d)$ if and only if

$$\text{dist}(n, d) < \text{dist}(n, s) + \text{dist}(s, d) \ . \tag{5.2}$$

[5] Our formal definitions (see later in this section) have been adapted by the newer version of the Remote LFA draft [130].

One can easily see, that (5.2) is the basic loop-free criterion of link-protecting LFAs [47], while (5.1) means that the repair tunnel cannot traverse the failed link.

The notion of $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ could also be expressed with distance functions:

Definition 8. *For a source s and next-hop e, some $n \in V$ is in the extended $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}(s,e)$ if and only if $\exists v \in neigh(s) : \mathrm{dist}(v,n) < \mathrm{dist}(v,s) + \mathrm{dist}(s,e) + \mathrm{dist}(e,n)$.*

It should be noted that the conditions above hold for arbitrary weighted graphs as well.

Next, we formulate an important corollary of the previous observations. In particular, we show that if an arbitrary node on the shortest path between a source and a destination is $\mathrm{rLFA_{LP}}$ protected, then every further node along that shortest path is $\mathrm{rLFA_{LP}}$ protected as well.

Lemma 1. *Let (s,d) be a source-destination pair and let q be a node along the default shortest path from s to d. If $\mathrm{rLFA_{LP}}(s,q) \neq \emptyset$, then $\mathrm{rLFA_{LP}}(s,d) \neq \emptyset$.*

Proof. Consider Fig. 5.3 and suppose node e is the next-hop from s to d. The wavy lines denote the existence of a path between the given nodes. The thick line indicates the shortest path from s to d ($s \to e \to q \to d$).

For n to be in $\mathrm{rLFA_{LP}}(s,d)$, it has to fulfill the conditions stated in Definition 7. First, it satisfies (5.1) for (s,d) since $\mathcal{P}_{\mathrm{LP}}$ does not depend on the destination node. Additionally, in case of link protection it only needs to satisfy (5.2), notably $\mathrm{dist}(n,d) < \mathrm{dist}(n,s) + \mathrm{dist}(s,d)$. We know that $\mathrm{dist}(n,q) < \mathrm{dist}(n,s) + dist(s,q)$ and due to the triangle inequality[6] $\mathrm{dist}(n,d) \leq \mathrm{dist}(n,q) + \mathrm{dist}(q,d)$. Therefore, $\mathrm{dist}(n,d) < \mathrm{dist}(n,s) + \mathrm{dist}(s,q) + \mathrm{dist}(q,d) \Rightarrow \mathrm{dist}(n,d) < \mathrm{dist}(n,s) + \mathrm{dist}(s,d)$. □

An important consequence of Lemma 1 is a simple observation that a graph has full $\mathrm{rLFA_{LP}}$ protection, if and only if each node has an $\mathrm{rLFA_{LP}}$ to each of its next-hops.

Corollary 2. *Let G be a graph with unit link costs. Then, $\mu(G) = 1$, if and only if for each $(u,v) \in E$, u has an $\mathrm{rLFA_{LP}}$ to v and v has an $\mathrm{rLFA_{LP}}$ to u.*

Next, we show that there is a deep connection between basic link-protecting LFA ($\mathrm{LFA_{LP}}$) and $\mathrm{rLFA_{LP}}$ in unit cost networks.

[6]The triangle inequality states that for any triangle, the sum of the lengths of any two sides must be greater than or equal to the length of the remaining side. It is one of the defining properties of the distance function, which is used in shortest path routing.

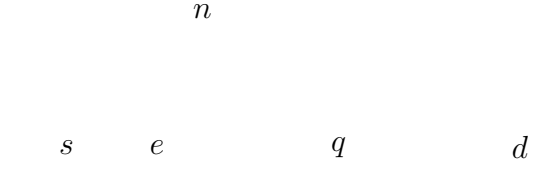


Figure 5.3. Illustration for the proof of Lemma 1

.

Theorem 5. *Let $G(V, E)$ be a graph with unit link costs, let (s, d) be a source-destination pair, let e be the default next-hop of s to d, and let u be an arbitrary node with $u \in \text{neigh}(s)$, $u \neq e$. Then, $u \in \text{rLFA}_{\text{LP}}(s, d)$ if and only if $u \in \text{LFA}_{\text{LP}}(s, d)$.*

Proof. First, we verify the forward direction. Easily, $u \in \text{rLFA}_{\text{LP}}(s, d)$ implies u is in $\mathcal{Q}_{\text{LP}}$, which precisely coincides with the condition for u to be a link-protecting LFA. Second, we check the reverse direction. If $u \in \text{LFA}_{\text{LP}}(s, d)$, then u, by definition, fulfills (5.2). In addition, it also satisfies (5.1) due to the assumption $u \in \text{neigh}(s)$, because in a uniform cost network the default shortest path between adjacent nodes is through the direct link, and hence the $s \to u$ shortest path always avoids the (s, e) link. □

Corollary 3. *Let $G(V, E)$ be a graph with arbitrary link costs, let (s, d) be a source-destination pair, let e be the default next-hop of s to d, and let u be an arbitrary node with $u \in \text{neigh}(s)$, $u \neq e$. Then, if $u \in \text{rLFA}_{\text{LP}}(s, d)$ then $u \in \text{LFA}_{\text{LP}}(s, d)$.*

As a proof, consider the forward direction of the proof above.

5.2.2 Node-protecting Case

In this section, we extend the previous statements to the case of node protection. Thus, suppose now that it is not the link between source s and its next-hop e that fails, but the next-hop e itself. It is easy to see from the above discussion that the condition for $\mathcal{P}_{\text{NP}}$ remains the same as for $\mathcal{P}_{\text{LP}}$, thus only $\mathcal{Q}_{\text{NP}}$ has to be re-defined.

Definition 9. *For a source s, next-hop e and destination d, some $n \in V$ is in $\mathcal{Q}_{\text{NP}}(s, d)$ if and only if*

$$\text{dist}(n, d) < \text{dist}(n, e) + \text{dist}(e, d) \ . \tag{5.3}$$

Similarly, it is easy to observe that (5.3) is the basic loop-free criterion of node-protecting LFAs [47]. The concept of $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}}$ could also be expressed as follows:

Definition 10. *For a source s and next-hop e, some $n \in V$ is in $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}}(s,e)$ if and only if $\exists v \in \mathrm{neigh}(s) : \mathrm{dist}(v,n) < \mathrm{dist}(v,e) + \mathrm{dist}(e,n)$.*

It should be noted again that these conditions also hold for arbitrary weighted graphs.

Next, we reformulate Lemma 1 and show that if an arbitrary node on the shortest path between a source and a destination is $\mathrm{rLFA}_{\mathrm{NP}}$ protected, then every further node along that shortest path is $\mathrm{rLFA}_{\mathrm{NP}}$ protected as well.

Lemma 2. *Let (s,d) be a source-destination pair and let q be a node along the default shortest path from s to d. If $\mathrm{rLFA}_{\mathrm{NP}}(s,q) \neq \emptyset$, then $\mathrm{rLFA}_{\mathrm{NP}}(s,d) \neq \emptyset$.*

Proof. Consider example network depicted in Fig. 5.3 again and suppose again node e is the next-hop from s to d. For n to be $n \in rLFA_{\mathrm{NP}}(s,d)$, it has to fulfill (5.1) and (5.3). As it was in the link-protecting case, we do not have to deal with $\mathcal{P}_{\mathrm{NP}}$ since it does not depend on the destination node. We only have to verify the condition of $\mathcal{Q}_{\mathrm{NP}}$: $\mathrm{dist}(n,d) < \mathrm{dist}(n,e) + \mathrm{dist}(e,d)$. Since $n \in \mathrm{rLFA}_{\mathrm{NP}}(s,q)$, then $\mathrm{dist}(n,q) < \mathrm{dist}(n,e) + \mathrm{dist}(e,q)$, and due to triangle inequality $\mathrm{dist}(n,d) \leq \mathrm{dist}(n,q) + \mathrm{dist}(q,d) \Rightarrow \mathrm{dist}(n,d) < \mathrm{dist}(n,e) + \mathrm{dist}(e,q) + \mathrm{dist}(q,d) \Rightarrow \mathrm{dist}(n,d) < \mathrm{dist}(n,e) + \mathrm{dist}(e,d)$, which completes the proof. □

Below, we show that, analogously to link protecting case, node-protecting LFAs and rLFAs are deeply related to each other in unit cost networks.

Theorem 6. *Let $G(V,E)$ be a graph with unit link costs, let (s,d) be a source-destination pair, let e be the default next-hop of s to d, and let u be an arbitrary node with $u \in \mathrm{neigh}(s)$, $u \neq e$. Then, $u \in \mathrm{rLFA}_{\mathrm{NP}}(s,d)$ if and only if $u \in \mathrm{LFA}_{\mathrm{NP}}(s,d)$.*

The proof of the theorem goes along similar lines as the proof of Theorem 5 so we do not present it herein. Furthermore, a similar corollary to Corollary 3 can be drawn for node protection as well.

5.3 Analysis of Extended Remote LFA

Next, we digress a little to show that extended rLFAs are a powerful tool for link-protection. In particular, first we show that in case of link failures, extended rLFA_{LP} ensures 100% failure coverage in every unit cost network.

Theorem 7. *Let G be an arbitrary 2-edge-connected graph with uniform link costs and suppose that link-protecting extended Remote LFA is deployed. Then, in case of link failures $\mu(G) = 1$.*

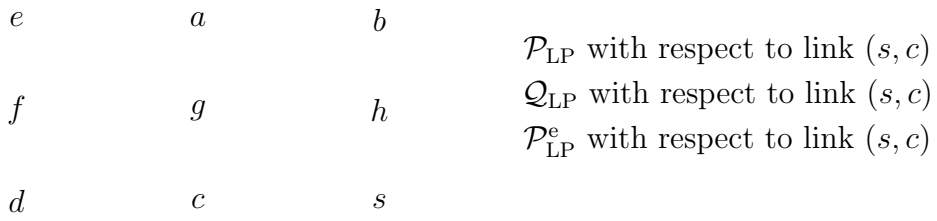
Proof. We show that for each edge $(u, v) \in E$, u has a Remote LFA to v (and vice versa). This will mean that every node has an rLFA_{LP} to each of its next-hops, which guarantees $\mu(G) = 1$ by Corollary 2. Since G is 2-edge-connected, we know that (u, v) is contained in at least one chordless cycle. Let the length of this cycle be k. If k is odd, then the single node at distance $\frac{k-1}{2}$ from v along the cycle is a Remote LFA to u. If, on the other hand, k is even, then the $\mathcal{P}_{\text{LP}}(u, (u, v)) \cap \mathcal{Q}_{\text{LP}}(u, v)$ is empty. Observe, however, that the single node of distance $\frac{k}{2}$ from u is contained both in $\mathcal{Q}_{\text{LP}}(u, v)$ and the extended $\mathcal{P}_{\text{LP}}(w, (u, v))$, where w is the neighbor of u other than v along the cycle, and so it is a link-protecting extended Remote LFA. This completes the proof. □

Consequently, in general it can be stated that the usage of extended rLFA provides full protection against single link failures in every, at least 2-edge-connected, unit cost network.

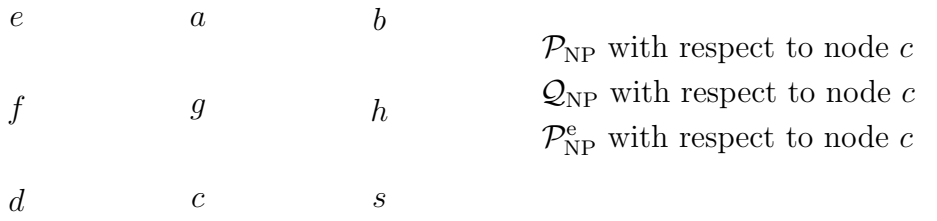
However, when node failures are also taken into account, then extended P-space with respect to the failed node is not always enough to guarantee 100% failure coverage in every case. The following theorem concludes this.

Theorem 8. *Let G be an arbitrary 2-edge-connected graph with uniform link costs and suppose that node-protecting extended Remote LFA is deployed. Then, in case of node failures, extended Remote LFA cannot guarantee complete protection.*

Proof. We show that there exists a simple network topology wherein not all node failures can be protected. Consider the network depicted in Fig. 5.4. Assume that node s wants to send a packet to node d. The default shortest path goes through node c. Under the assumption that only the link (s, c) fails (see Fig. 5.4(a)), then



(a) $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ provide full rLFA protection against single link failures



(b) $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}}$ cannot guarantee full protection against single node failures

Figure 5.4. Illustration of a network with different protection scenarios

it can be protected since $\mathcal{P}_{\mathrm{LP}} \cap \mathcal{Q}_{\mathrm{LP}} \neq \emptyset$ ($b \in \mathrm{PQ}_{\mathrm{LP}}$-nodes). Next, consider the case of Fig. 5.4(b) where the next-hop c went down. The potential repair tunnel endpoints are in $\mathrm{PQ}_{\mathrm{NP}}$-nodes, which is again the intersection of $\mathcal{P}_{\mathrm{NP}}$ of s and $\mathcal{Q}_{\mathrm{NP}}$ of d. Unfortunately, this remains $\emptyset$ even if using the extended P-space $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ would be an option. Observe that since node a is not directly connected to the failed node c, de facto node-protecting could also not be assumed to resolve this issue. This means that there are networks that cannot be 100% protected against node failures by nor "plain" neither extended Remote LFA. □

From the above discussion, one can easily see that the requirements of node protection are stricter than those for link protection. We can summarize our observations as follows.

Lemma 3. $\mathcal{P}_{\mathrm{LP}} = \mathcal{P}_{\mathrm{NP}}$ *but* $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}} \subseteq \mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$, *and* $\mathcal{Q}_{\mathrm{NP}} \subseteq \mathcal{Q}_{\mathrm{LP}}$.

Proof. First, note that it was already concluded that the protection scheme does not affect P-spaces. Second, to prove the connection between $\mathcal{P}^{\mathrm{e}}_{\mathrm{LP}}$ and $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}}$ we use Definition 8. and Definition 10. In the case of $\mathcal{P}^{\mathrm{e}}_{\mathrm{NP}}$, there is a node n: $\mathrm{dist}(v, n) < \mathrm{dist}(v, e) + \mathrm{dist}(e, n)$, where $v \in \mathrm{neigh}(s)$ and e is the default next-hop. Due to triangle

inequality $\text{dist}(v,e) \leq \text{dist}(v,s) + \text{dist}(s,e)$, and using this in our formal definition of $\mathcal{P}^{\text{e}}_{\text{NP}}$ results in that $\text{dist}(v,n) < \text{dist}(v,e)+\text{dist}(e,n) \leq \text{dist}(v,s)+\text{dist}(s,e)+\text{dist}(e,n)$, which corresponds to the formal definition of $\mathcal{P}^{\text{e}}_{\text{LP}}$. Therefore, $\mathcal{P}^{\text{e}}_{\text{NP}} \subseteq \mathcal{P}^{\text{e}}_{\text{LP}}$. Third, since $\mathcal{Q}_{\text{LP}}$ and $\mathcal{Q}_{\text{NP}}$ are actually the loop-free criteria of link- and node-protecting LFAs, respectively, the property $\mathcal{Q}_{\text{NP}} \subseteq \mathcal{Q}_{\text{LP}}$ is inherited from pure LFA. Note again that these observations are hold for arbitrary link costs. □

5.4 Summary

In Chapter 4, we showed that pure LFA cannot protect every single network failure. We discussed that improvements can be made by altering the existing network topology. If modifying the network is not an option, deploying Remote LFAs may be a better approach. Therefore, in this chapter we deeply analyzed Remote LFA in both link- and node-protecting cases. As in the case of LFA, the number of failure cases protected by rLFA crucially depends on both the graph topology and the link costs. As it seems difficult to consider both at the same time, we studied graph topological concerns separately from the effect of link costs.

We analyzed rLFA failure coverage in general networks by a new set of elemental graph theoretical rLFA tools. Moreover, we extended the basic specification of rLFA [80], originally defined for single link failures only, to the relevant case of node failures, and we also made a deep analysis to this case with our toolset.

We showed that, under the unit-cost assumption, "extended P-space" results in full rLFA failure coverage in every network against single link failures. This can be an important pointer for operators, currently in the position to deploy rLFA, on how to actually choose link costs. Unfortunately, it turned out that in the case of node protection this option is not enough to protect all source-destination pairs.

In the next chapter, we continue our analysis and study what topological properties bound the performance of rLFA. Furthermore, as a possible way of improvement, we define rLFA Graph Extension problem as the task to augment an unweighted graph with the fewest new links to obtain 100% rLFA failure case coverage.

Chapter 6

Lower Bounds and Network Optimization Algorithms for Remote LFA

So far, we have seen that Remote LFA can definitely provide higher failure case coverage than pure LFA. However, it was shown that full protection can still not be guaranteed even in a simple network due to its inherited topological dependence. Therefore, in this chapter we give a graph-theoretical characterization of rLFA coverage, as measured by $\mu_{\mathrm{LP}}(G)$ and $\mu_{\mathrm{NP}}(G)$. Our main aim is to identify the attainable *lower and upper bounds* of plain rLFA failure coverage against both link and node failures in *unit cost* networks. We describe some methods to easily calculate failure coverages in different families of notable graph topologies. As it was in Chapter 5, first, we deal with single link failures, then we focus on node protection as well.

As shall be shown, there are some networks in which Remote LFA failure coverage is particularly low. Therefore, our second goal is to define special network optimization algorithms, purposed at attaining full rLFA coverage in every network by augmenting the network with suitably chosen new links.

6.1 Graphs with Good Remote LFA Coverage

Below, we return to the case of plain Remote LFAs and we suppose that only this variant is available in seeking possible repair tunnel endpoints. Hence, in the rest of

this Dissertation, we consider only the standard definitions for P-spaces and Q-spaces.

Network operators facing with the challenge of deploying Remote LFA need to ask the question, whether their current network topology is amenable to rLFA or not. Therefore, it is crucial to separate graph topologies that are "good" for rLFA (i.e., the ones with $\mu(G) = 1$) away from those that attain a particularly low coverage.

6.1.1 Link-protecting Case

First, we characterize graphs that are highly rLFA-protected against link failures, and then we extend our analysis to the node-protecting case. The following theorem states the requirements for a network to be full rLFA$_{LP}$ protected.

Theorem 9. *Let G be an undirected, simple graph with uniform link costs. Now, $\mu(G) = 1$, if and only if for each $(i,j) \in E : \exists n \neq i,j$ so that* $\text{dist}(i,n) = dist(j,n)$.

Proof. The result comes from applying (5.1) and (5.2) directly to (i,j). Therefore, $\mathcal{P}_{\text{LP}}$ can be defined as $\text{dist}(i,n) < \text{dist}(i,j) + \text{dist}(j,n)$, while $\mathcal{Q}_{\text{LP}}$ as $\text{dist}(j,n) < \text{dist}(n,i) + \text{dist}(i,j)$. Since link costs are unit cost, then $\text{dist}(i,j) = 1$, accordingly $\text{dist}(i,n) < 1 + \text{dist}(j,n)$ and $\text{dist}(j,n) < \text{dist}(n,i) + 1 \rightarrow \text{dist}(j,n) - 1 < \text{dist}(i,n) < \text{dist}(j,n) + 1 \Rightarrow \text{dist}(i,n) = \text{dist}(j,n)$. The backward direction of the proof comes from Cor. 2. □

There are numerous graph topologies known from graph theory, where rLFA$_{LP}$ failure coverage is 100% such as chordal graphs [131] (see Fig. 6.2(d)) or "Möbius ladder" topologies with k even (see Fig. 6.2(c)). For instance, in chordal graphs with uniform link costs every node is in a triangle, which means that if a link between an arbitrary node pair (s,d) fails, then the third node (n) in any of the triangles consisting of both s and d is the neighbor of s and d, respectively, i.e., $dist(s,n) = dist(d,n)$. 'Möbius ladder" topologies depicted in Fig. 6.2(c) are fully protected in case of k even, since due to the links connecting the left and right side of the network, each node is contained in an odd cycle. Therefore, for an arbitrary source-destination pair (i,j) there always exists a node n for which the distance from i is equal to the distance from j. However, if k is odd, then each node is contained in an even cycle, which is not fully protected. See formal proofs later.

6.1.2 Node-protecting Case

Next, we characterize the good cases for rLFA$_{NP}$ and show that, as it was in the link-protecting case, there exist graphs that can be fully protected against single node failures. Since here, node protection is considered as undefined between two arbitrary neighboring nodes, we need to analyze only those (s, d) pairs for which $\text{dist}(s, d) > 1$ applies[1]. The following theorem concludes the results:

Theorem 10. *Let G be an undirected, simple graph with uniform link costs, and let $\mathcal{S}_2$ be a set of* 2-neighbors *in G: $(u, v) : \text{dist}(u, v) = 2$. Now, $\mu_{\text{NP}} = 1$, if and only if for each $(s, d) \in \mathcal{S}_2$ source-destination pairs and e next-hop there exists $n \neq s, d, e$ for which*

$$\text{dist}(s, n) = \text{dist}(n, d) \text{ or } \text{dist}(s, n) + 1 = \text{dist}(n, d) \ .$$

Proof. Consider the (s, d) pair in $\mathcal{S}_2$ depicted in Fig. 6.1, where $\text{dist}(s, d) = 2$ and the wavy lines denote the existence of paths among the nodes. In this case, for n to be $\text{rLFA}_{\text{NP}}(s, d)$ it has to fulfill (5.1) and (5.3), namely $\text{dist}(s, n) < 1 + \text{dist}(e, n)$ and $\text{dist}(n, d) < \text{dist}(e, n) + 1 \Rightarrow \text{dist}(s, n) = \text{dist}(n, d)$. On the other hand, consider now that $\text{dist}(e, n) = k$. Then, due to triangle inequality, $\text{dist}(s, n)$ and $\text{dist}(n, d)$ can only be $1 \leq x \leq k$. However, if $x = 1$, then $\text{dist}(n, d)$ can only be $\text{dist}(s, n) + 1$, since if it does not, then the next-hop of s to destination d would not be node e. The backward direction of the proof comes from Lemma 2, as if a next-hop is rLFA_{NP} protected, then every further node, including the next-next-hop, is protected as well. Therefore, if it is true for each non-neighboring node pair, then $\mu_{\text{NP}}(G) = 1$. □

There is a bunch of networks for which the statement of Theorem 10 applies. For instance, infinite grids and "Möbius ladder" topologies with arbitrary k all qualify.

Note that if fallback to link protection was supported, the requirement for a network to be full rLFA protected against single node failures change as follows:

Corollary 4. *Let G be an undirected, simple graph with uniform link costs, and let $\mathcal{S}_2$ be a set of* 2-neighbors *in G: $(u, v) : \text{dist}(u, v) = 2$. Now, $\mu_{\text{NP}} = 1$, if and only if for each $(s, d) \in \mathcal{S}_2$ source-destination pairs and e next-hop there exists $n \neq s, d, e$ for which $\text{dist}(s, n) = \text{dist}(n, d)$ or $\text{dist}(s, n) + 1 = \text{dist}(n, d)$, and for each directly connected (s, d), there exists $n \neq s, d$ so that $\text{dist}(s, n) = \text{dist}(d, n)$.*

[1]Since only unit cost networks are considered, $\text{dist}(s, d) > 1$ means that node d is at least 2 hops away from node s and vice versa.

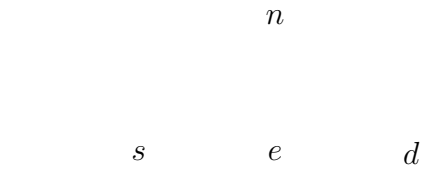


Figure 6.1. Illustration for Theorem 10

Assuming this protection strategy, the previously mentioned infinite grids are not protected regardless of whether k is odd or even; and similarly to link-protecting case "Möbius ladder" topologies are only protected if k is even.

6.2 Lower Bounds

In the following, we turn to discuss the *lower bounds* for rLFA, that is, we seek worst-case graphs, whose coverages against single link failures are particularly poor.

6.2.1 Link-protecting Case

It has been discussed in Chapter 3 that quintessential worst-case graphs for IPFRR are rings. Consequently, we consider odd rings first, and then we shall treat even rings.

Next, we generalize Proposion 1 to rLFA_{LP}. In fact, we shall do a bit more, as our analysis will account for the length of the repair tunnel, which is an important factor in provisioning Remote LFA[2].

Theorem 11. *Let C_n be an odd ring on n nodes with $n \geq 3$, and let $1 \leq k \leq \frac{n-1}{2}$ denote an upper bound on the length of the tunnel from the source node to its* rLFA. *Then, $\mu(C_n) = \frac{2k}{n-1}$.*

Proof. Consider a ring topology on n nodes, n odd, let $(s, d) \in E$ be a *neighboring* source-destination pair and suppose that the link between them went down. In this case, s needs to find a possible remote loop-free alternate since it cannot use its other neighbor because it will pass back the packet. Thus, the possible repair tunnel

[2]See the `remote-lfa maximum-cost` option on [81]

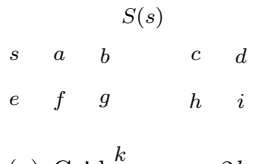
endpoints are situated on the other side of the ring with respect to the failed link, i.e., if an arbitrary node $u \in \text{rLFA}_{\text{LP}}(s,d)$, then $\text{dist}(s,u) \leq \frac{n-1}{2}$ which is tight if $d \in \text{neigh}(s)$. One can observe that if maximal tunnel length is permitted, i.e., $k = \frac{n-1}{2}$, then such kind of repairing node always exists ($\mu_{\text{LP}}(C_n) = \frac{n(n-1)}{n(n-1)} = 1$). However, if the tunnels need to be shorter than an arbitrary node u can only be an rLFA_{LP} is $\text{dist}(s,u) \leq \frac{n-1}{2} - l$, where l is the tunnel shortening coefficient, i.e., the greater the l, the shorter the tunnel. Trivially, shortening the tunnel with l dissolves the protection among $\forall(s,d)$ pairs, where $dist(s,d) = l$. Therefore, rLFA_{LP} failure coverage can be modified as follows: $\mu_{\text{LP}}(C_n) = \frac{n(n-1-2l)}{n(n-1)}$. Now, consider $\text{dist}(s,u) \leq k$, where k represents the length of the tunnel. In this manner $l = \frac{n-1}{2} - k$ meaning that $\mu_{\text{LP}}(C_n) = \frac{n-1-n+1+2k}{n-1} = \frac{2k}{n-1}$. □

Note that $k = 1$ means that only neighboring nodes can be used as repair tunnel endpoints, which essentially corresponds to the case of simple loop-free alternates. For this particular case, Theorem 11 yields the same result as Prop. 2 stated for LFA_{LP} for odd rings.

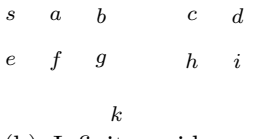
Theorem 12. *Leg C_n be an even ring on n nodes with $n \geq 4$, and let $1 \leq k \leq \frac{n-2}{2}$ denote an upper bound on the length of the tunnel from the source node to its* rLFA_{LP}. *Then, $\mu_{\text{LP}}(G) = \frac{2k-1}{n-1}$.*

Proof. Consider a ring on n nodes, n even, and suppose that link between an arbitrary *neighboring* (s,d) source-destination pair went down. According to the case of odd ring, s needs to pass the packet to the other side of the ring, however, the possible repair tunnel endpoints cannot be reached without traversing the failed component. Thus, for $\forall(s,d)$ pairs, where $d \in neigh(s)$: the link (s,d) cannot be protected. One can observe, if $\text{dist}(s,d) \geq 2$, then tunnels, avoiding the failed link exist. Therefore, for an arbitrary source s has Remote LFAs to $\forall d$ destination excluding its neighbors ($\mu_{\text{LP}}(C_n) = \frac{n(n-3)}{n(n-1)}$). However, assuming shorter tunnels results that for possible $u \in \text{rLFA}_{\text{LP}}(s,d) : \text{dist}(s,u) \leq \frac{n}{2} - l$, where l is a shortening coefficient as it was in the case of odd rings. Now, $l = \frac{n}{2} - k$ meaning that $\mu_{\text{LP}}(C_n) = \frac{n-1-n+2k}{n-1} = \frac{2k-1}{n-1}$. □

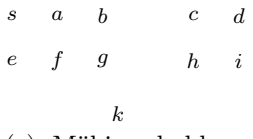
As before, supposing $k = 1$ results the corresponding statement in Prop. 1 for LFA_{LP} for even rings. In this regard, rLFA_{LP} can be seen as a natural generalization of LFA_{LP}.



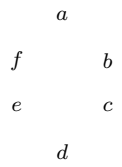
(a) Grid on $n = 2k$ nodes



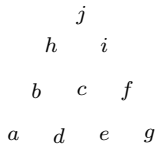
(b) Infinite grid on $n = 2k$ nodes



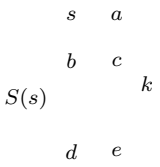
(c) Möbius ladder on $n = 2k$ nodes



(d) Chordal graph



(e) 4-Propeller graph on $n = 3k + 1$



(f) Complete bipartite graph on $n = 2k$ nodes

Figure 6.2. Illustration topologies

Worst-case Scenarios: 2-node-connected Graphs

Below, we continue our analysis towards finding 2-node-connected graphs with low rLFA_{LP} failure coverage. In what follows, we suppose that there is no constraint on the length of the tunnel.

Since the simplest 2-node-connected network with low failure coverage is a 4-cycle ($\mu_{\text{LP}}(C_4) = \frac{1}{3}$), we examined graphs that contain a large number of 4-cycles as subgraphs. We considered the networks depicted in Fig. 6.2(a) where k denotes the number of "columns", and Fig. 6.2(f) where k marks the number of node pairs. The following theorem concludes the results:

Theorem 13. *For any $k > 2$ there is a 2-node-connected graph G on $n = 2k$ nodes with $\mu_{\text{LP}}(G) = \frac{k-1}{2k-1} < 0.5$.*

Proof. Next, we show that grids (G_k) and complete bipartite graphs ($K_{k,k}$) attain this limit. In grids, $\forall(s, d)$ pairs: $d \in \text{neigh}(s)$ or $d \in \text{S}(s)$ cannot be protected, where $\text{S}(s)$ denotes the set of nodes situated on the same side. It is easy to see that every node is in a 4-cycle wherein neighbors as destinations are not protectable and the shortest paths to every node on the same side traverse one of the neighbors. Thus, such nodes are unprotected according to Lemma 1.

Similar is the case for $K_{k,k}$ as well. Each $d \in \mathrm{S}(s)$ are protected while $\forall d' \notin \mathrm{S}(s)$ are neighbors of s and, due to the property of bipartite graphs that every cycle is even, neighbors cannot be protected either. □

Worst-case Scenarios: 2-edge-connected Graphs

So far, we have seen that in 2-node-connected graphs as many as 50% of the node-pairs can go unprotected by rLFA against single link failures. Below, we show that in slightly less dense 2-edge-connected graphs the situation can be even worse.

Theorem 14. *For any $k \geq 1$ there is a 2-edge-connected graph G on $n = 3k + 1$ nodes with $\mu_{\mathrm{LP}}(G) = \frac{1}{3}$.*

Proof. Here, we show that the so called "4-propeller graph" (P_k) attains this limit, where k denotes the number of blades. Thus, consider (P_3) depicted in Fig. 6.2(e). One can see that the nodes on the pitch of the propeller blades have Remote LFAs to every destination except the neighbors, since they are on an even cycle. Nodes on the side of the blades considered as sources can only protect adjacent link failures if the nodes in the face of them are considered as destinations. Finally, the node in the middle has Remote LFAs only for destination nodes situated on the pitch of the blades. Thus,

$$\mu_{\mathrm{LP}}(G) = \frac{k(3k-2) + 2k + k}{3k(3k+1)} = \frac{3k^2 + k}{3k(3k+1)} = \frac{k(3k+1)}{3k(3k+1)} = \frac{1}{3} \ .$$

□

6.2.2 Node-protecting Case

Next, we turn to find the graphs with the lowest $\mathrm{rLFA}_{\mathrm{NP}}$ coverage, as measured by μ_{NP}, moreover we show that even $\mu_{\mathrm{NP}}(G) = 0$ is possible, and this can be attained even in a not so complicated network topology. As mentioned above, there are certain graphs for which studying μ_{NP} does not make any sense, as it happens to be undefined. Such is the case, for instance, of complete graphs with unit link costs: here every node-pair is adjacent and hence we consider rLFA_{NP} as undefined due to the *last-hop*

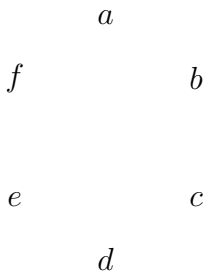


Figure 6.3. Worst-case graph for rLFA_{NP} on $n = 6$ nodes

problem[3]. Therefore, we only contemplate graphs in which at least one non-adjacent node pair exists (i.e., non-complete graphs). Even in these graphs, the question is only interesting when single node failures, at least theoretically, can be repaired, so we need to focus only on *2-node-connected graphs*.

The following theorem concludes the results, wherein again node protection between neighboring nodes are considered undefined:

Theorem 15. *For any $n > 4$, there is a 2-node-connected graph G on n nodes with $\mu_{\text{NP}}(G) = \frac{2(n-3)}{n^2-5n+6} \leq \frac{4}{n}$.*

Proof. We show a particular graph on n nodes, hereafter denoted by $\mathcal{L}_n$, that attains this limit. An example for $\mathcal{L}_n$ for the case when $n = 6$ is depicted in Fig. 6.3. It is easy to see that the chordal graph on Fig. 6.2(d) (G_{c_6}) and $\mathcal{L}_6$ are not isomorphic with each other. Furthermore, the failure coverages also differ, since $\mu_{\text{NP}}(G_{c_6}) = \frac{2}{3}$ and $\mu_{\text{NP}}(\mathcal{L}_6) = \frac{1}{2}$.

The main topological characteristic of $\mathcal{L}_n$ is that there is one node on the top with degree of $n-1$, there are two nodes with degree of 2, while the remaining $n-3$ nodes have a degree of 3. Correspondingly, the number of non-adjacent source-destination pairs is $2(n-3) + (n-3)(n-4) = n^2 - 5n + 6$. For each non-neighboring node pair (s, d) : $\text{dist}(s, d) = 2$ via the node on the top. One easily sees, in addition, that only those node pairs can be protected that have ECMPs to each other, that is, which are in opposite in the 4-cycles. The number of such node pairs equals twice the number of 4-cycles in the graph (i.e., $n-3$), and therefore there are $2(n-3)$ protected node pairs. Consequently, we have $\mu_{\text{NP}}(\mathcal{L}_n) = \frac{2(n-3)}{n^2-5n+6}$. Observe that, in

[3]Note that if fallback to link protection was supported, in complete graphs only link-protecting rLFAs would considered.

the limit, this bound tends to zero, meaning that in very large $\mathcal{L}_n$ graphs the fraction of rLFA node-protected source-destination pairs diminishes. □

Note that in case of fallback to link protection, the number of protected source-destination pairs increases by $2(n-1)+2(n-2)$, since there are $2(n-1)$ protected neighboring links that connect the node on the top with every other node, while there are $2(n-2)$ remaining protected neighboring links on the outer cycle. In this case, the above-mentioned rLFA coverage changes to $\mu_{NP}(G) = \frac{2(n-3)+2(n-1)+2(n-2)}{n(n-1)}$ resulting in higher rLFA coverage, however it is still strictly less than $\frac{6}{n}$. Thus, in very large L_n graphs, node-protecting rLFA coverage with capability to fallback to link protection diminishes as well. In the rest of this Dissertation, rLFA_{NP} is considered as *undefined* between neighboring source-destination pairs.

So far, we have sought a tight characterization for the lower bounds on μ_{LP} and μ_{NP} for any unweighted graph G. At the moment, we do not have clear answers to this intriguing but hard graph-theoretical problem. What we could prove, however, is that in certain 2-node-connected unweighted graphs $\mu_{\text{LP}}(G)$ can be as low as $\frac{1}{2}$, and in 2-edge-connected graphs an even lower threshold of $\frac{1}{3}$ is also realizable. So far, we have not been able to identify any 2-node-connected or 2-edge-connected graph with smaller rLFA_{LP} coverage. Thus, we conjecture that $\frac{k-1}{2k-1}$ is an actual lower bound on $\mu_{\text{LP}}(G)$ for 2-node-connected unit costs graphs, while $\frac{1}{3}$ is a lower bound on $\mu_{\text{LP}}(G)$ for 2-edge-connected unit cost graphs. In the case of node protection, we have stronger results: we could show that there exist certain large graphs with $\mu_{\text{NP}} \to 0$, which, evidently, is a lower bound, at least in the limit. Regarding graphs of practical size, however, we do not have better lower bound at the moment than the one for $\mathcal{L}_n$ graphs.

6.2.3 Computational Study

It turned out that finding a universal lower bound on rLFA_{LP} or rLFA_{NP} coverage is a hard problem. Clearly, a computational approach might be instructive to support or refute our conjectures. Hence, we generated all non-isomorphic networks on $n < 10$ nodes. Note that the generation is very time consuming even if only non-isomorphic graphs are created. Table 6.1 summarizes the lower bounds with the following notations: n denotes the number of nodes, μ_{LP}^{2e} and μ_{LP}^{2n} note failure coverages against

Table 6.1. Lower bounds measured by μ_{LP} and μ_{NP} in worst-case graphs on n nodes

n	μ_{LP}^{2e}	B_l	μ_{LP}^{2n}	B_l	μ_{NP}	B_l
3	1	1	1	1	undefined	undefined
4	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1	1
5	$\frac{2}{5}$		$\frac{2}{5}$		$\frac{2}{3}$	$\frac{2}{3}$
6	$\frac{2}{5}$		$\frac{2}{5}$	$\frac{2}{5}$	$\frac{1}{2}$	$\frac{1}{2}$
7	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{3}{7}$		$\frac{2}{5}$	$\frac{2}{5}$
8	$\frac{19}{56}$		$\frac{3}{7}$	$\frac{3}{7}$	$\frac{1}{3}$	$\frac{1}{3}$
9	$\frac{1}{3}$		$\frac{31}{72}$		$\frac{2}{7}$	$\frac{2}{7}$

single link failures in case of 2-edge-connected and 2-node-connected networks, respectively. The failure case coverage against single node failures in non-complete 2-node-connected networks is denoted by μ_{NP}. Columns marked by B_l denote the conjectural lower bounds. In case of link protection, it can be seen that until $n \leq 4$ results are the same, and if $n \geq 5$ coverages start to increase. One can observe that in the case of $n = 7$ the given failure coverage equals the coverage attained by 4-propeller graphs mentioned above. It also shows that lower bounds of 2-edge-connected networks are the lowest.

The case of node protection turns out different. If $n = 4$ the coverage is 1, while if $n > 4$ the coverages start to decrease. The most important observation is that the conjectured lower bounds are tight for every 2-node-connected network until $n = 9$ nodes suggesting that the result of Theorem 15 hold true as the attainable lower bound on μ_{NP} in 2-node-connected networks. Moreover, the networks we found as conjectural lower bounds (marked by B_l) and the generated networks that provide the practical lower bounds were isomorphic. This suggests that besides the networks we found for the given number of nodes, there is no other non-isomorphic network that could attain the same lower bounds.

The measured results and the deviation compared to the protection provided by pure LFA are summarized in Table 6.2.

Table 6.2. Lower bounds provided by LFAs in different network topologies, where n marks the number of nodes, while in case of 2-node-connected networks and $\mu_{LP}(G)$ $2k = n$

LFA type	2-edge-connected	2-node-connected
$\eta_{LP}(G)$	$\frac{1}{n-1}$	$\frac{1}{n-1}$
$\mu_{LP}(G)$	$\frac{1}{3}$	$< \frac{k-1}{2k-1}$
$\eta_{NP}(G)$	undefined	$\frac{1}{n-1}$
$\mu_{NP}(G)$	undefined	$< \frac{4}{n}$

6.3 Remote LFA Graph Extension

As observed above, there exist a lot of graphs with small failure coverage, measured in terms of μ_{LP} and μ_{NP}. Hence, in this section, we ask to what extent we need to intervene at the graph topology to improve rLFA coverage to 100% in both link- and node-protecting cases. This problem is important since *(i)* this would answer how "far" are poorly protected networks from perfect rLFA failure coverage and *(ii)* would provide an easy way for operators to boost the protection in their networks. We adapt the formal description of the LFA Graph Extension problem from [84] to link-protecting rLFA as follows:

Definition 11. *Link-protecting rLFA Graph Extension problem (rLFAGraphExtensionLP(G)): Given a graph $G(V, E)$, find the smallest subset F of the complement edge set $\overline{E}$ of G such that $\mu_{\mathrm{LP}}(G(V, E \cup F) = 1$.*

Similarly, in the case of node protection, this definition can be formalized as follows:

Definition 12. *Node-protecting rLFA Graph Extension problem (rLFAGraphExtensionNP(G)): Given a graph $G(V, E)$, find the smallest subset F of the complement edge set $\overline{E}$ of G such that $\mu_{\mathrm{NP}}(G(V, E \cup F) = 1$.*

Note that in both cases the cost of the new link is of unit cost as well, therefore at least on shortest path will change.

Since all previously studied LFA network optimization problems turned out NP-complete and so there were no optimal algorithms that could solve the problems in viable time, we immediately turned to approximation algorithms in this case as well.

To actually solve the problem, we adopted the *greedy Graph Extension algorithm* from [84], which, at least for LFA, performed almost the same as the optimal algorithm, but it is much faster and simpler. Here, we extend this algorithm to the case of rLFA, both for the link-protecting and the node-protecting cases. Moreover, we also developed a *simulated annealing-based heuristic* as another approach to complement our studies in increasing the rLFA failure coverage in different networks.

First, we show the greedy Graph Extension method. This algorithm adds the best link from the complement link set that improves the coverage at most. Formally, the algorithm is defined as follows:

Algorithm 2 Greedy rLFA Graph Extension for graph $G(V, E)$

1: **while** $\mu(G(V, E)) < 1$
2: $\quad (u, v) \leftarrow \underset{(i,j)\in\overline{E}}{\operatorname{argmax}}\ \mu(G(V, E \cup \{(i, j)\}))$
3: $\quad E \leftarrow E \cup \{(i, j)\}$
4: **end while**

Note that the pseudo-code works the same for the link-protecting and the node-protecting cases. The variant for the link-protecting case is called the *Greedy Link-Protecting rLFA Graph Extension algorithm*, while the one, optimizing for node-protection, is called the *Greedy Node-Protecting rLFA Graph Extension algorithm*.

The following theorems characterize the terminating conditions of these algorithms.

Theorem 16. *Let $G(V, E)$ be a graph with unit link costs. Then, the Greedy Link-protecting rLFA Graph Extension algorithm terminates with full link-protecting rLFA coverage regardless of the input graph.*

Proof. Alg. 2 surely terminates when all complement links are added, but at this point $\mu_{\mathrm{LP}}(G) = 1$ as complete graphs have full link-protecting rLFA coverage. □

Theorem 17. *Let $G(V, E)$ be a graph with unit link costs and suppose that $G(V, E)$ is not a complete graph. Then, the Greedy Node-Protecting rLFA Graph Extension algorithm terminates with full node-protecting rLFA coverage regardless of the input graph.*

Proof. We cannot use the same approach directly as above, because with all the complement links added we again reach a complete graph but for this graph μ_{NP} is

considered as undefined. We observe, however, that if we add all the complement links except one, then we get an almost complete graph in which node protection is defined between one and only node pair. As this node pair is trivially protected against a single node failure (since the nodes in the pair are not neighbors and they are situated in a 4-cycle), therefore $\mu_{\mathrm{NP}} = 1$ for this graph. As the algorithm is guaranteed to converge to this graph, unless the input is a complete graph or the algorithm terminates previously, the proof is complete. □

Next, we turn to the other algorithm. Similarly to LFA Cost Optimization, we chose the simulated annealing probabilistic metaheuristic as the main framework, and within this framework we obtained different heuristics by different objective functions. Basically, the algorithm works as follows: given an input graph $G(V, E)$, we try to augment the graph with a randomly chosen link (i, j) from the complement link set $\overline{E}$. If the failure coverage was improved, then we unconditionally accept this edge. Otherwise, the link could still be accepted with a certain probability ensuring the system to not get stuck in local optima. The process terminates at that time when temperature is dropped to 0 or failure coverage reached 1. As it was in case of LFA Cost Optimization, we also used *tabu lists* to preclude the iteration from oscillating between two or more already tested new links.

Algorithm 3 Simulated Annealing based rLFA Graph Extension for graph $G(V, E)$

```
1: T ← T_0
2: while μ(G(V,E)) < 1
3:     choose_random_link((i,j) ∈ Ē)
4:     if accept_link(Δμ, T) then
5:         E ← E ∪ {(i,j)}
6:     end if
7:     T ← T − 1
8: end while
```

The pseudo-code for the simulated annealing based heuristic is given in Alg. 3. Note again, that it works similarly for the link-protecting and node-protecting case. The pseudo-code uses two procedures, specified as follows:

- choose_random_link(i, j) selects randomly an edge $(i, j) \in \overline{E}$ to be added to the network.

- accept_link($\Delta\mu, T$): after adding a randomly selected new link to the network, the new failure coverage $\mu(G(V, E \cup (i, j)))$ is examined. If it was improved, then the new link is added irrevocably to the network. Otherwise, two different objective functions (ΔD) are evaluated in order to check how bad the new solution is. One of them only checks to what extent the failure coverage declined, formally $\Delta D = \mu(G(V, E)) - \mu(G(V, E \cup (i, j)))$. Besides, the other objective function takes into account the number of newly added links as well in order to try to keep it low, formally: $\Delta D = \Delta NP - 2m$, where NP is the number of protected source-destination pairs and m is the number of all the link including the newly added one as well. These objective functions are tested via metropolis test as it was discussed in Section 4.3.4. Again, metropolis test is used in simulated annealing heuristics to accept "bad" solutions if it suits for a criterion, namely $e^{(\frac{-\Delta D}{T})} > R(0, 1)$, where ΔD is the change of the objective function, T is the actual temperature of the system, and $R(0, 1)$ is a random number in the interval $[0, 1]$. According to the output of this test, the newly added link is left in the network permanently or promptly removed. One can observe that the two different objective functions result in two different kinds of heuristics. Therefore, let SA_{co} be the simulated annealing algorithm with the former objective function, and let SA_{cne} be the simulated annealing algorithm with the latter objective function.

6.3.1 Numerical Evaluations

In this section, we examine how many links one must add in realistic graphs to achieve full rLFA coverage, both against single link- and node-failures.

In all topologies, we set link costs uniformly to 1. Note that there are networks in the data set where inferred link costs were exactly unit costs.

We executed the greedy algorithm as well as the simulated annealing based heuristics. From the latter we executed 20000 rounds, with initial temperature $T_0 = 150$ and tabu list size of 20. The detailed results of the link-protecting case are in Table 6.3 with the following notations: η_{LP} is the initial link-protecting LFA coverage; μ_{LP} is the initial link-protecting rLFA coverage; Gr_η denotes the number of new links added by the LFA Greedy Graph Extension algorithm to reach 100% link-protecting

Table 6.3. Remote LFA Graph Extension results for link protection

Topology	η_{LP}	Gr_η	μ_{LP}	Gr_μ	$SA_{\Delta\mu}$	SA_γ
AS1221	0.833	1	0.833	1	1	1
AS1239	0.898	6	1	0	0	0
AS1755	0.889	4	1	0	0	0
AS3257	0.946	3	0.954	1	1	1
AS3967	0.864	7	0.969	1	1	1
AS6461	0.919	2	1	0	0	0
Abilene	0.56	6	0.833	1	1	1
Arnes	0.595	18	0.731	6	9	12
AT&T	0.823	6	0.8875	2	2	2
Deltacom	0.542	79	0.885	4	7	11
Gambia	0.037	16	0.111	8	12	13
Geant	0.646	20	0.827	4	5	5
Germ_50	0.801	22	1	0	0	0
Germany	0.695	1	0.882	1	1	1
InternetMCI	0.877	3	0.888	2	2	2
Italy	0.784	12	0.951	2	2	2
NSF	0.86	9	1	0	0	0

LFA coverage, while Gr_μ gives the same results for Remote LFA. $SA_{\Delta\mu}$ denotes the number of new links added by SA_{co}, and last but not least, SA_γ marks the number of new links added by SA_{cne}.

The first observation is that there were five networks that were fully protected with rLFA right away, even without the need of any graph extension. Second, the number of links that have to be added to reach full coverage with rLFA is much less than when only simple link-protecting LFA capable routers are present, irrespectively of which graph extension method was used. Nevertheless, on average the number of links added by the different simulated annealing based heuristics is greater than in the case of the greedy algorithm. This suggests that for the Remote LFA Graph Extension problem the greedy approach is the most plausible solution and, if we can draw conclusions from the case of pure LFA in [84], it probably performs very close to the optimal solution too. The largest improvement in rLFA coverage, compared to simple LFA, is seen in networks where initially $\eta(G) < 0.9$ (see, e.g., in the Geant topology). In the Deltacom topology, the installation of 79 new links was necessary

to achieve full LFA coverage, while with only 4 additional links full rLFA coverage is attainable. The results indicate that *(i)* more than 50% of the networks lend themselves to rLFA Graph Extension since the maximum number of links needed is less than 2; *(ii)* on average 3.6 new links are necessary to attain 100% rLFA coverage while in case of simple LFA this number is 14.5.

In the second run, we examined how the proposed algorithms could improve the failure coverage against single node failures. Since the extended rLFA variant can play an important role in the case of node protection, even if the link costs are uniform, we evaluated that possibility as well[4]. Table 6.4 contains the results, where η_{NP} is the initial node-protecting LFA coverage; μ_{NP} is the initial node-protecting rLFA coverage, while μ^e_{NP} is the initial node-protecting extended rLFA coverage. Gr_η denotes the number of new links added by the LFA Greedy Graph Extension algorithm, Gr_μ marks the number of new links added by the rLFA Greedy Graph Extension algorithm, while in the case of Gr^e_μ the extended P-space option was also considered. The results in column $SA_{\Delta\mu}$ and SA_γ are similar to the link-protecting case, while columns $SA^e_{\Delta\mu}$ and SA^e_γ indicates the number of links added by the two simulated annealing based heuristics, under the assumption that extended rLFA is used for providing protection.

The first observation is that, if simple rLFA is considered then there is no network, which is initially fully rLFA protected against node failures. However, in case of Extended rLFA, there were 3 networks with full protection out of the box. As it was in the link-protecting case, much less additional links are needed for 100% node-protecting rLFA failure coverage than when only simple node-protecting LFAs are only available. For instance, in Deltacom topology, 113 new links were necessary to protect all source-destination pairs with pure LFA against single node failures, while this number is only 9 when Remote LFAs can be used as well. One can also observe that the greedy approach yielded the best solutions, i.e., it needed the fewest additional links in order to provide full protection. Namely, in the case of plain rLFA, on average it installs 4.05 new links to the network, while simulated annealing based algorithms could not reach full protection with less than 6.35 new links. Nevertheless, if extended P-space is taken into account, then greedy algorithm needed on average

[4]Note that as mentioned above extended rLFA provides full failure case coverage against single link failures in every unit cost network.

Table 6.4. Remote LFA Graph Extension results for node protection

Topology	η_{NP}	Gr	μ_{NP}	Gr	$SA_{\Delta\mu}$	SA_{γ}	μ^{e}_{NP}	Gr^{e}_{μ}	$SA^{e}_{\Delta\mu}$	SA^{e}_{γ}
AS1221	0.083	3	0.083	1	1	1	0.083	1	1	1
AS1239	0.658	16	0.843	1	1	1	0.928	1	1	1
AS1755	0.704	7	0.912	1	1	1	1	0	0	0
AS3257	0.521	20	0.702	5	8	8	0.866	3	3	3
AS3967	0.715	10	0.896	2	2	2	0.994	1	1	1
AS6461	0.505	8	0.596	3	3	3	0.747	2	2	2
Abilene	0.608	3	0.725	2	2	2	0.872	1	1	1
Arnes	0.331	35	0.426	12	24	20	0.45	12	16	15
AT&T	0.565	12	0.684	4	4	4	0.849	2	2	2
Deltacom	0.436	113	0.818	9	25	27	0.868	9	22	23
Gambia	0.04	23	0.12	14	17	22	0.12	13	19	18
Geant	0.411	30	0.676	5	11	11	0.74	5	8	8
Germ_50	0.676	37	0.977	1	1	1	0.998	1	1	1
Germany	0.599	8	0.77	2	2	2	0.955	2	2	2
InternetMCI	0.558	9	0.837	3	2	2	0.916	1	1	1
Italy	0.574	24	0.839	3	3	3	0.926	2	2	2
NSF	0.634	16	0.963	1	1	1	1	0	0	0

3.3 new links, whilst the other two heuristics resulted in more than 4.75 new links.

As it was mentioned in Section 4.3.5, in certain cases when the network topology changes for longer period because of manual interventions, the execution times of the different algorithms could play an important role. In Fig. 6.4, the average execution times of the different algorithms are depicted from left to right in a descending order of the attained LFA coverage. The leftmost Gr_{μ} denotes the execution time for the greedy approach, while $SA_{\Delta\mu}$ and SA_{γ} mark the execution times for the different simulated annealing based heuristics. On average, both of the heuristics were running 4 minutes longer than the greedy approach, and the heuristic denoted by $SA_{\Delta\mu}$ was, on average, 8 seconds faster than the other one (SA_{γ}). Note that the average time the greedy algorithm required for completion was *only* 0.5 seconds, thus it is not just the fastest algorithm but it also outperforms the other proposed heuristics.

Overall, the results suggest that network operators might hugely benefit from deploying rLFA in their routers, since it can definitely protect much more source-destination pairs than pure LFA ever could do. Moreover, the provisioning of a very

Figure 6.4. Execution times for the greedy algorithm and the different heuristics in descending order from left to right according to their attained LFA coverage

small number of additional new links can boost the protection provided by rLFA up to 100%. In particular, we found that more than 50% of the networks needed less than 4 additional links for perfect rLFA failure case coverage.

6.4 Summary

In this chapter, we studied general lower and upper bounds for rLFA coverage. For upper bounds, we showed that in both link- and node-protecting cases, full rLFA coverage can be attained. For lower bounds, in the case of link protection, we found that for 2-node-connected graphs on $2k$ nodes the value $\frac{k-1}{2k-1}$ is realizable by grids and complete bipartite graphs, and we confirmed computationally that this is a valid lower bound as long as the number of nodes n is smaller than 10. We also found that for 2-edge-connected graphs, this "conjectured" lower bound is $\frac{1}{3}$. We showed that for node failures rLFA coverage can, somewhat unexpectedly, straight out drop to zero in certain cases.

As a way of improvement, we defined the rLFA Graph Extension problem as the task to augment an unweighted graph with the fewest new links to obtain 100% failure case coverage. Along a simple greedy algorithm, we also developed a family of simulated annealing based heuristics to solve this problem approximately. We found that, as it was in the case for pure LFA [84], the greedy method is the most plausible

algorithm. It turned out that, even in very big real-world ISP topologies, adding only $2-3$ new links is enough to attain 100% failure coverage against link failures, whilst the number of new links needed for full protection against node failures is only slightly more, $3-4$.

Chapter 7

Conclusion

Currently, Loop-Free Alternates is the best choice for providing fast protection in pure IP and MPLS/LDP networks, as it is readily implemented and available in basically all commercial IP router offerings. It is a well known fact that LFA cannot protect every single failure, therefore many network operators are currently hesitating whether to enable this protection mechanism. As a possible improvement, a network optimization method, called LFA Graph Extension, was proposed, which tries to effectively augment the network topology with suitable chosen new links in order to improve the low LFA protections reachable for sake of its topological dependence. However, this might not be an universal solution for every network operator, since in certain cases the number of new links that have to be added may not be afforded. Therefore, in this Dissertation we sought further possible ways to assist network operators in making this important decision.

In particular, in Chapter 4 we showed to what extent the LFA failure case coverage can be improved by optimizing link costs instead. Besides the fact that transient link failures are the most common failures in an operational network, we also took into account the relevant case of single node failures. We defined LFA Cost Optimization problem and studied it in huge detail. We showed that this problem is NP-complete, and we gave an exact algorithm of exponential complexity as well as a family of efficient heuristics with tunable performance and running time to solve it. Our selection of heuristics facilitated for picking the right approximation algorithm for the particular problem under consideration. Furthermore, we provided a comprehensive numerical evaluation of LFA Cost Optimization methods to compare

their performance on a wide range of artificial and real-world network topologies. We found that a significant boost in LFA coverage in many real-world network topologies can be achieved by merely optimizing link costs, and what is more, in some cases close to perfect protection could be guaranteed by LFA.

However, in [98] it was found that in sparser topologies the highest attainable LFA coverage can be particularly poor. Therefore, we discussed to what extent the protection can be improved by an efficient combination of LFA Graph Extension and LFA Cost Optimization.

This combined approach is called LFA Combined Optimization, and we showed that among all optimization methods, this produces the best results as it reduces on average by more than 50% the number of additional links formerly was necessary to achieve 100% LFA coverage by LFA Graph Extension.

Recently, as an extension, the IETF has published a generalization of LFA, called the Remote LFA IP Fast ReRoute Framework (rLFA), which can provide additional backup connectivity when none can be provided by pure LFA. Unfortunately, its failure case coverage still depends on the underlying network topology leaving network operators still in doubt, since so far, there has been no information available about its performance, fundamental lower and upper bounds on failure case coverage, or even how this could be improved.

In Chapter 5, we developed a set of elemental graph theoretical rLFA tools, which facilitate for analyzing rLFA failure coverage in general networks. Similarly to our LFA analysis, we extended the rLFA specification [80], originally defined for single link failures only, to single node failures, and we generalized our toolset to this very case as well. However, we slightly diverged from the specification, and we defined “plain” and “extended rLFA”. Using our toolset, we provided a comprehensive analysis of rLFA failure case coverage under the assumption that network links are of unit costs. Furthermore, we revealed deep relations between pure LFA and rLFA, and we showed the conclusions that can be drawn if information about one of them exists.

We showed that extended rLFA can provide full protection against single link failures in a unit cost network. This can be an important pointer for operators, currently in the position to deploy rLFA, on how to actually choose link costs. Unfortunately, it turned out that in the case of node protection this option is not enough to protect all source-destination pairs.

In Chapter 6, we gave sufficient and necessary conditions for full rLFA failure coverage in the case of single link as well as node outages. Furthermore, an attempt was made to find lower bounds on failure case coverage provided by rLFA. In particular, we found that in 2-node-connected graphs rLFA protection for single link failures can go down to 50%, or to 33% for 2-edge-connected networks, and for node failures rLFA coverage can practically zero out in certain cases. To help inherently poorly protected networks, we adapted the LFA Graph Extension approach, and studied the rLFA Graph Extension problem in detail. This problem asks to augment the network with new, unit cost links to attain complete rLFA protection. To solve this problem, we proposed a complete family of heuristics in order to facilitate for picking the best approximation algorithm for the particular network under consideration. Similarly to our LFA analysis, we also provided an extensive numerical evaluation of rLFA failure case coverage and rLFA Graph Extension methods on a wide range of real-world network topologies. Crucially, we found that some networks have full rLFA protection without any modifications. For the rest, the proposed heuristics turned out very effective in improving rLFA failure protection.

7.1 Application of Results

We believe that our results may have a significant impact not just in an academic setting but for the industry as well. We hope that our results may contribute to the general understanding of the failure case coverage provided by LFAs, and may help hesitating network operators to reach a verdict of using any of the available approaches. In particular, our research in LFA based optimization was conducted with Ericsson Research, Hungary, TrafficLab as an industrial partner, and our optimization algorithms were implemented and used in an internal application with graphical user interface (see Fig. 7.1), in which network operators can analyze and optimize their network(s).

Regarding our Remote LFA analysis, we have been in contact with the IETF to make Remote LFA Draft as an RFC[1]. Our alternative definitions of P-,Q- and extended P-spaces in terms of costs (Eq. 5.1, Eq. 5.2 and Eq. 8) provided an easier

[1]The Draft is in its final state (11^{th} version, last call), which means that if there is no need for a major revision until August 3, 2015, then it will become an RFC.

Figure 7.1. A screenshot of our LFAAnalyser application

understanding in the current version of the draft [80], since in the specification of LFA [47], the basic loop-free conditions were also defined in terms of costs. Moreover, we believe that analyzing Remote LFA first in the literature and showing its advantages and network optimization approaches will give an increasing push on service providers to operate the Internet without any interruption and indeed win the trust of most of the potential users.

Nevertheless, it should be noted that several router vendors have already implemented Remote LFA in their newer devices, and it could be enabled by a couple of commands [132, 133].

7.2 Theses Summary

In this section, I summarize the main results of this dissertation and I present *my theses*.

Thesis 1. *I have defined the LFA Cost Optimization problem formally. I have proven that this problem is NP-complete both in the link-protecting and node-protecting cases,*

and I have proposed a heuristic framework to obtain approximate solutions. By extensive numerical evaluations, I have found that my algorithms yield at least 10% *improvement in LFA coverage on many real-world network topologies (see Section 4.3 in Chapter 4).*

Thesis 1.1. *[J2, C1] For any $k > 0$ integer, there is a graph G on $n = 4k + 2$ nodes with two cost functions c_1 and c_2, so that $|\eta(G, c_1) - \eta(G, c_2)| \geq \frac{1}{2}$.*

Thesis 1.2. *[J2, C1] I have proven that LFACostOptLP(G, $\mathcal{S}$) is NP-complete (see Theorem 3).*

Thesis 1.3. *[J2] I have shown that LFA Cost Optimization problem remains NP-complete in the node-protecting case (see Theorem 2).*

Thesis 1.4. *[J2, C1] I have developed a family of fast and efficient heuristics for solving the LFACostOptLP(G, $\mathcal{S}$) problem. A series of extensive simulation studies I have conducted on real and artificial network topologies suggest that on average at least* 10% *LFA coverage improvement can be attained in real-world network topologies (see Section 4.3.4 and 4.3.5).*

Thesis 1.5. *[J2] I have extended the algorithmic framework to cover the node-protecting case as well, and I have conducted simulations indicating that the algorithms increase LFA protection against single node failures by* 10-20% *(see Section 4.3.4 and 4.3.5).*

Thesis 2. *I have defined the LFA Combined Network Optimization problem formally. I have proven that this problem is NP-complete in both the link-protecting and node-protecting cases, and I have shown how the LFA Graph Extension and the LFA Cost Optimization methods should be combined. I have also given simulations indicating that both adding new links to a network and optimizing link costs at the same time are effective ways to improve the LFA coverage in operational networks (see Section 4.4 in Chapter 4).*

Thesis 2.1. *[J1] I have shown that the Combined LFA Network Optimization problem is NP-complete (see Theorem 4).*

Thesis 2.2. *[J1] I have shown that the combined algorithm significantly reduces (on average by more than* 50%*) the number of additional links necessary for reaching* 100% *LFA coverage (see Section 4.4.1).*

Thesis 3. *I have performed analytical and numerical study of the level of protection achievable by Remote LFA in both link- and node-protecting cases. I have given alternative sufficient and necessary conditions for a node pair to be rLFA-protected in an arbitrary network, I have found a deep connection between basic LFAs and Remote LFAs, and I have proven that in unit cost networks extended Remote LFA provides* 100% *failure case coverage against single link failures. I have also proven that this statement does not apply to node protection (see Chapter 5).*

Thesis 3.1. *[C2, J3] For source s, destination d, and next-hop e, some node $n \neq s, d$ is a link-protecting Remote LFA for the $s-d$ pair (i.e., $n \in \mathrm{rLFA_{LP}}(s,d)$) if and only if*

$$\mathrm{dist}(s,n) < \mathrm{dist}(s,e) + \mathrm{dist}(e,n) \tag{7.1}$$

$$\mathrm{dist}(n,d) < \mathrm{dist}(n,s) + \mathrm{dist}(s,d) \ . \tag{7.2}$$

This is defined in Definition 7.

Thesis 3.2. *[C2, J3] For source s, destination d, and next-hop e, some node $n \neq s, d$ is an extended link-protecting Remote LFA for the $s-d$ pair if and only if*

$$\exists v \in neigh(s) : \mathrm{dist}(v,n) < \mathrm{dist}(v,s) + \mathrm{dist}(s,e) + \mathrm{dist}(e,n) \tag{7.3}$$

$$\mathrm{dist}(n,d) < \mathrm{dist}(n,s) + \mathrm{dist}(s,d) \ . \tag{7.4}$$

This is defined in Definition 8.

Thesis 3.3. *[C2, J3] I have proven the following equivalence conditions for basic LFA and rLFA in both link- and node-protecting cases:*

- *For an arbitrary node $u \in \mathrm{rLFA}(s,d)$ and $u \in \mathrm{neigh}(s) \Rightarrow u \in \mathrm{LFA}(s,d)$, where $\mathrm{neigh}(s)$ is the set of nodes which are directly connected to node s (see Theorem 5).*
- *For an arbitrary node $u \in \mathrm{rLFA}(s,d)$ and $u \in \mathrm{neigh}(s) \Leftrightarrow u \in \mathrm{LFA}(s,d)$ under the assumption that costs are uniform (see Corollary 3).*

Thesis 3.4. *[J3] For source s, destination d, and next-hop e, some $n \neq s, d$ is a*

node-protecting Remote LFA for the $s-d$ pair (i.e., $n \in \mathrm{rLFA}_{\mathrm{NP}}(s,d)$) if and only if

$$\mathrm{dist}(s,n) < \mathrm{dist}(s,e) + \mathrm{dist}(e,n) \tag{7.5}$$

$$\mathrm{dist}(n,d) < \mathrm{dist}(n,e) + \mathrm{dist}(e,d) \ . \tag{7.6}$$

This is defined in Definition 9.

Thesis 3.5. *[J3] For source s, destination d, and next-hop e, some node $n \neq s,d$ is an extended node-protecting Remote LFA for the $s-d$ pair if and only if*

$$\exists v \in \mathrm{neigh}(s) : \mathrm{dist}(v,n) < \mathrm{dist}(v,e) + \mathrm{dist}(e,n) \tag{7.7}$$

$$\mathrm{dist}(n,d) < \mathrm{dist}(n,e) + \mathrm{dist}(e,d) \ . \tag{7.8}$$

This is defined in Definition 10.

Thesis 3.6. *[C2, J3] I have shown that extended Remote LFA provides complete protection against single link failures in any unit cost network (see Theorem 7).*

Thesis 3.7. *[J3] I have shown that, in general, extended Remote LFA does not necessarily provide complete protection against single node failures, not even in unit cost networks (see Theorem 8).*

Thesis 4. *I have found that in certain unit cost networks Remote LFA coverage can be as low as 33% for single link failures, and 0% for single node failures. I have also proposed heuristic algorithms to approximately solve the rLFA Graph Extension problem for both single link and node failures. I have given extensive simulations indicating that in the case of link protection on average 3.6 new links are enough to attain 100% rLFA coverage, in case of node protection 4.05 links are necessary, while for extended node-protecting rLFA this number is 3.3 (see Chapter 6).*

Thesis 4.1. *[C2, J3] I have shown that for any $k \geq 1$ there is a 2-edge-connected graph G on $n = 3k+1$ nodes with $\mu(G) = \frac{1}{3}$ (see Theorem 14).*

Thesis 4.2. *[C2, J3] I have found that for any $k > 2$ there is a 2-node-connected graph G on $n = 2k$ nodes with $\mu_{\mathrm{LP}}(G) = \frac{k-1}{2k-1} < 0.5$ (see Theorem 13).*

Thesis 4.3. *[J3] I have found that for any $n \geq 4$, there is a 2-node-connected graph G on n nodes with $\mu_{\mathrm{NP}}(G) = \frac{2(n-3)}{n^2-5n+6} \leq \frac{4}{n}$ (see Theorem 15).*

Thesis 4.4. *[C2, J3] I have developed a family of fast and efficient heuristics for solving $rLFAGraphExtension(G)$ problem. By extensive numerical evaluations, I have found that for more than 50% of the examined unit cost networks only 2 new links are needed to attain 100% $\mathrm{rLFA}_{\mathrm{LP}}$ coverage, and on average the number of new links is only 3.6. In the node-protecting case, on average 4.05 additional links are necessary, which reduces to 3.3 with extended rLFA (see Section 6.3).*

7.3 Future Work

Since LFA does not require any protocol changes, it can be deployed router-by-router without shutting down and restarting all network elements throughout the network. However, in an operational network operators may enable LFA only at some critical points, where a possible failure has a huge impact in traffic loss and/or it does not affect significantly the link utilization. Consequently, it may be worth to customize the algorithms presented in this Dissertation for these cases, where not all source-destination pairs need to be protected or some source-destination pairs have higher priority than others, i.e., the cases when only certain shortest paths must be retained but others can change arbitrarily; or the costs are optimized both for LFA coverage *and* to provide efficient utilization of the network with respect to the expected traffic matrices.

In case of our Remote LFA analysis, in the future we plan to study the case of unweighted networks as well as further Remote LFA related network optimization questions should be examined. For instance, similarly to LFA Cost Optimization, improving rLFA coverage is possible with modifying link costs as well, which looks another intriguing, and practically relevant and fair network optimization problem.

Recently, the field of Software Defined Networks (SDN) has arisen, which makes it possible to decouple the control plane and forwarding plane of an IP router. It is not just a theoretical approach, since nowadays many router vendors are producing SDN enabled devices. This means that one can write a simple application as a control logic and can easily test its behavior in a real network, without having to rely on simulations. So far, large-scale testbeds have been deployed [OC3, OC4] in order to study new and different methods in real-world networks. With this in mind, the performance of different IPFRR methods can be examined in a realistic manner,

which was not possible until now. Therefore, in the future we plan to put to use not just our results, but all the approaches emerged so far in the field of reliable networks by means of Software Defined Networks [OC5].

Publications

Journal Papers

[J1] **L. Csikor**, M. Nagy, and G. Rétvári, "Network Optimization Techniques for Improving Fast IP-level Resilience with Loop-Free Alternates," Infocommunications Journal, vol. 3, iss. 4, pp. 2-10, 2011. ($6/2 = 3$)

[J2] **L. Csikor**, G. Rétvári, and J. Tapolcai, "Optimizing IGP Link Costs for Improving IP-level Resilience with Loop-Free Alternates," Computer Communications Journal, Special Issue on Reliable Network-based Services, vol. 36, iss. 6, pp. 645-655, 2013. ($6/2 = 3$)

[J3] **L. Csikor** and G. Rétvári, "On Providing Fast Protection with Remote Loop-Free Alternates," Telecommunication Systems Journal, vol. 60, iss. 4, pp. 485-502, 2015. ($6/1 = 6$)

Conference Papers

[C1] G. Rétvári, **L. Csikor**, J. Tapolcai, G. Enyedi, and A. Császár, "Optimizing IGP Link Costs for Improving IP-level Resilience," in Proc. International Workshop on Design Of Reliable Communication Networks (DRCN), Krakow, Poland, pp. 62-69, 2011. (winner of Best Paper Award) ($3/4 = 0.75$)

[C2] **L. Csikor** and G. Rétvári, "IP Fast Reroute with Remote Loop-Free Alternates: the Unit Link Cost Case," in Proc. RNDM, pp. 16-22, 2012. ($3/1 = 3$)

Book Chapters

[B1] **L. Csikor**, G. Rétvári and J. Tapolcai. "High Availability in the Future Internet", Future Internet Assembly 2013: Validated Results and New Horizons,

Lecture Notes in Computer Science, *The Future Internet*, vol. 7859, pp. 64-76, 2013. (6/2 = 3)

Other Publications

[OC2] **L. Csikor** and Z. Fehér. "Exploring Hidden Relations in Moving Human Groups". In *Proc., POSTER 2010*, Prague, Czech Republic, 6. May 2010. (3/2 = 1.5)

[OC1] **L. Csikor** and Z. Fehér. "Information Spreading in Self Oragnizing Mobile Networks". In *Proc., MACRo Conference 2010*, Tirgu Mures, Romania, 14-15. May 2010. (3/2 = 1.5)

[OJ1] P. Babarczi, F. Tanai, **L. Csikor**, J. Tapolcai and Z. Heszberger. "Útvonalválasztás késleltetés-toleráns hálózatokban". *Híradástechnika*, vol. 66, no. 1, pp. 23-31, 2011. (1/5 = 0.2)

[OC3] F. Németh, B. Sonkoly, A. Gulyás, **L. Csikor**, J. Tapolcai, P. Babarczi and G. Rétvári. "Improving resiliency and throughput of transport networks with OpenFlow and Multipath TCP: Demonstration of results over the Géant OpenFlow testbed". *Open Networking Summit*, Flyer and Demo, Santa Clara, USA, Apr. 2013.

[OC4] F. Németh, B. Sonkoly, **L. Csikor**, and A. Gulyás, "A Large-Scale Multipath Playground for Experimenters and Early Adopters," in ACM SIGCOMM (DEMO), Hong Kong, China, pp. 482-483, 2013.

[OC5] B. Sonkoly, F. Németh, **L. Csikor**, L. Gulyás, and A. Gulyás, "SDN based testbeds for evaluating and promoting multipath TCP," in Proc. IEEE International Conference on Communications (ICC), pp. 3044-3050, June 2014. (3/5 = 0.6)

[OC6] A. Csoma, B. Sonkoly, **L. Csikor**, F. Németh, A. Gulyás, W. Tavernier and S. Sahhaf "ESCAPE: Extensible Service ChAin Prototyping Environment using Mininet, Click, NETCONF and POX". In *Proc., ACM SIGCOMM 2014*, Demo, Chicago, Aug. 2014.

[OC7] A. Csoma, B. Sonkoly, **L. Csikor**, F. Németh, A. Gulyás, D. Jocha, J. Elek, W. Tavernier and S. Sahhaf "Multi-layered Service Orchestration in a Multi-Domain Network Environment". In *Proc., EWSDN 2014*, Demo, Budapest, Sep. 2014.

Bibliography

[1] G. Sallai, "Defining infocommunications and related terms," *Acta Polytechnica Hungarica*, vol. 9, no. 6, pp. 5–15, 2012.

[2] Cisco, "Cisco visual networking index: Forecast and methodology, 2012–2017," White Paper, May 29, 2013.

[3] ITU-T, "ICT facts and figures," http://www.itu.int/ITU-D/ict/facts/2011/material/ICTFactsFigures2010.pdf (downloaded: Jan. 2013), 2011.

[4] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2012-2017," White Paper, February 6, 2013.

[5] D. R. Kuhn, "Sources of failure in the public switched telephone networks," *IEEE Computer*, vol. 30, no. 4, pp. 31–36, April 1997.

[6] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," RFC 3031, 2001.

[7] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5th ed. Prentice Hall, 2011.

[8] J. Moy, "OSPF version 2," RFC 2328, Apr. 1998.

[9] ISO, "Intermediate ststem-to-intermediate system (is-is) routing protocol," ISO/IEC 10589, 2002.

[10] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," RFC 4271, 2006.

[11] F. Baker, "Requirements for IP version 4 routers," RFC 1812, June 1995.

[12] Cisco Systems, "Cisco ios release 12.0 and network protocols configuration guide," 2011.

[13] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[14] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, "LDP specifiaction," RFC 5036, Oct. 2007.

[15] L. Andersson, I. Minei, and B. Thomas, "LDP specifiaction," RFC 5036, Oct. 2007.

[16] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta, "MPLS label stack encoding," RFC 3032, Jan. 2001.

[17] C. Systems, "MPLS - label assignment and distribution," https://www.racf.bnl.gov/Facility/TechnologyMeeting/Archive/06-30-04-CISCO/MPLS-Label-Assignment-Distribution.pdf, 2002.

[18] W. Simpson, "IP in IP tunneling," RFC 1853, October 1995.

[19] Cisco Systems, "Bidirectional forwarding detection for ospf," White Paper, 2005.

[20] D. Katz and D. Ward, "Bidirectional forwarding detection (BFD)," RFC 5880, June 2010.

[21] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet routing instability," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 515–528, 1998.

[22] S. Iyer, S. Bhattacharyya, N. Taft, and C. Diot, "An approach to alleviate link overload as observed on an IP backbone," in *INFOCOM*, 2003.

[23] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP networks," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 3, pp. 35–44, 2005.

[24] C. Systems, "Enhanced interior gateway routing protocol," White Paper, Document ID: 16406, September 09, 2005.

[25] D. Savage, D. Slice, J. Ng, S. Moore, and R. White, "Enhanced interior gateway routing protocol," Internet Draft, October, 2014.

[26] M. B. Doar, "A better model for generating test a useful simpli cation introduced by tiers to networks," in *Globecom*, November 1996.

[27] K. Calvert, M. B. Doar, and E. W. Zegura, "Modeling internet topology," *IEEE Communications Magazine*, vol. 35, pp. 160–163, 1997.

[28] M. Balakrishnan and A. Reibman, "Characterizing a lumping heuristic for a markov network reliability model," in *FTCS*, 1993, pp. 56–65.

[29] V. D. Park and M. S. Corson, "A performance comparison of the temporally-ordered routing algorithm and ideal link-state routing," in *ISCC*, 1998, pp. 592–598.

[30] A. U. Shankar, C. Alaettinoglu, I. Matta, and K. Dussa-Zieger, "Performance comparison of routing protocols using MaRS: Distance vector versus link-state," *ACM SIGMETRICS Performance Evaluation Review*, vol. 20, no. 1, pp. 181–192, June 1992.

[31] V. Paxson, "End-to-end routing behavior in the internet," in *SIGCOMM*, 1996, pp. 25–38.

[32] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental study of Internet stability and backbone failures," in *FTCS*, 1999, pp. 278–285.

[33] N. Feamster and H. Balakrishnan, "Detecting bgp configuration faults with static analysis," in *NSDI*, 2005, pp. 43–56.

[34] W. Xu, L. Huang, A. Fox, D. Paterson, and M. Jordan, "Detecting large-scale system problems by mining console logs," in *SOSP*, 2009.

[35] A. Shaikh, C. Isett, A. Greenberg, M. Roughan, and J. Gottlieb, "A case study of OSPF behavior in a large enterprise network," in *IMC*, 2002, pp. 217–230.

[36] D. Watson, F. Jahanian, and C. Labovitz, "Experiences with monitoring OSPF on a regional service provider network," in *ICDCS*, 2003, pp. 204–212.

[37] A. Markopoulou, G. Iannacone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *Proc. IEEE Infocom*, Mar. 2004.

[38] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, "Characterization of failures in an operational IP backbone network," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 749–762, 2008.

[39] G. Iannaccone, C.-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," in *ACM SIGCOMM Internet Measurement Workshop*, 2002, pp. 237–242.

[40] B. Choi, S. Song, G. Koffler, and D. Medhi, "Outage analysis of a university campus network," in *16th International Conference on Computer Communication and Networks (ICCCN)*, August 2007, pp. 204–212.

[41] A. Medem, R. Teixeira, N. Feamster, and M. Meulle, "Determining the causes of intradomain routing changes," University Pierre and Marie Curie, Tech. Rep., 2009.

[42] D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California Fault Lines: Understanding the causes and impact of network failures," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 315–326, August 2010.

[43] C. Boutremans, G. Iannacconne, and C. Diot, "Impact of link failures on VoIP performance," in *of NOSSDAV workshop, ACM press*, May 2002.

[44] M. Shand and S. Bryant, "IP Fast Reroute framework," RFC 5714, Jan 2010.

[45] J. P. G. Sterbenz, E. K. Cetinkaya, M. A. Hameed, A. Jabbar, S. Qian, and J. P. Rohrer, "Evaluation of network resilience, survivability , and disruption tolerance: Analysis, topology generation, simulation and experimentation," *Springer Telecommunication Systems Journal*, pp. 1–32, 2011.

[46] J.-P. Vasseur, M. Pickavet, and P. Demeester, "Network recovery: Protection and restoration of optical, SONET-SDH, IP, and MPLS," The Morgan Kaufmann Series in Networking, 2004.

[47] A. Atlas and A. Zinin, "Basic specification for IP fast reroute: Loop-Free Alternates," RFC 5286, 2008.

[48] A. Atlas, "U-turn alternates for IP/LDP fast-reroute," Internet-Draft, draft-atlas-ip-local-protect-uturn-03, February 2006.

[49] S. Bryant, C. Filfils, S. Previdi, and M. Shand, "IP fast reroute using tunnels," Internet-Draft, draft-bryant-ipfrr-tunnels-03, Nov. 2007.

[50] J. Lau, M. Townsley, and I. Goyret, "Layer two tunneling protocol - version 3 (L2TPv3)," RFC 3931, March 2004.

[51] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, "Generic routing encapsulation (GRE)," RFC 2784, March 2000.

[52] P. Francois, "Improving the convergence of IP routing protocols," Ph.D. Thesis, biblio.info.ucl.ac.be/2007/457147.pdf, 2007.

[53] G. Schollmeier, J. Charzinski, A. Kirstädter, C. Reichert, K. Schrodi, Y. Glickman, and C. Winkler, "Improving the resilience in IP networks," in *Proc. HPSR*, 2003.

[54] S. Nelakuditi, S. Lee, Y. Yu, and Z.-L. Zhang, "Failure insensitive routing for ensuring service availability," in *Proceedings International Workshopon Quality of Service (IWQoS)*, 2003.

[55] S. Lee, Y. Yu, S. Nelakuditi, Z.-L. Zhang, and C.-N. Chuah, "Proactive vs reactive approaches to failure resilient routing," in *INFOCOM*, 2004.

[56] Z. Zhong, S. Nelakuditi, Y. Yu, S. Lee, J. Wang, and C.-N. Chuah, "Failure inferencing based fast rerouting for handling transient link and node failures," in *INFOCOM*, 2005.

[57] J. Wang and S. Nelakuditi, "IP fast reroute with failure inferencing," in *ACM SIGCOMM Workshop on Internet Network Management – The Five-Nines Workshop*, 2007.

[58] J. Wang, Z. Zhong, and S. Nelakuditi, "Handling multiple network failures through interface specific forwarding," in *IEEE Globecom*, November 2006.

[59] G. Enyedi, G. Rétvári, and T. Cinkler, "A novel loop-free IP fast reroute algorithm," in *EUNICE*, 2007.

[60] G. Enyedi and G. Rétvári, "A loop-free interface-based fast reroute technique," in *4th EURO-NGI Conf. on Next Generation Internet Networks (EuroNGI)*, Cracow, Poland, 2008, pp. 29–44.

[61] G. Enyedi, "Novel algorithms for IP fast reroute," Ph.D. Thesis, 2011.

[62] S. Bryant, M. Shand, and S. Previdi, "IP fast reroute using Not-via addresses," Internet-Draft, March 2010.

[63] A. Li, P. Francois, and X. Yang, "On improving the efficiency and manageability of NotVia," in *ACM CoNEXT*, 2007.

[64] G. Enyedi, P. Szilágyi, G. Rétvári, and A. Császár, "IP Fast ReRoute: lightweight Not-Via without additional addresses," in *INFOCOM Mini-conf*, 2009.

[65] A. Kvalbein, A. F. Hansen, T. Čičic, S. Gjessing, and O. Lysne, "Fast IP network recovery using multiple routing configurations," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, 2006, pp. 1–11.

[66] A. Kvalbein, A. F. Hansen, T. Čičic, S. Gjessing, and O. Lysne, "Multiple routing configurations for fast IP network recovery," *IEEE/ACM Transactions on Networking*, vol. 17, no. 2, pp. 473–486, 2009.

[67] T. Čičic, A. F. Hansen, A. Kvalbein, M. Hartmann, R. Martin, M. Menth, S. Gjessing, and O. Lysne, "Relaxed multiple routing configurations: Ip fast

reroute for single and correlated failures," *Network and Service Management, IEEE Transactions on*, vol. 6, no. 1, pp. 1–14, 2009.

[68] P. Merindol, J.-J. Pansiot, and S. Cateloin, "Providing protection and restoration with distributed multipath routing," in *Performance Evaluation of Computer and Telecommunication Systems, 2008. SPECTS 2008. International Symposium on*, june 2008, pp. 456–463.

[69] K. Lakshminarayanan, M. Caesar, M. Rangan, T. Anderson, S. Shenker, and I. Stoica, "Achieving convergence-free routing using failure-carrying packets," in *Proc. SIGCOMM*, 2007.

[70] I. Hokelek, M. Fecko, P. Gurung, S. Samtani, S. Cevher, and J. Sucec, "Loop-free IP fast reroute using local and remote LFAPs," Internet-Draft, Feb 2008.

[71] A. Császár, G. Enyedi, J. Tantsura, S. Kini, J. Sucec, and S. Das, "IP fast reroute with fast notification," Internet-Draft, draft-csaszar-rtgwg-ipfrr-fn-01.txt, February 2013.

[72] K.-W. Kwong, L. Gao, R. Guerin, and Z.-L. Zhang, "On the feasibility and efficacy of protection routing in IP networks," in *INFOCOM, long version is available in Tech. Rep. 2009*, University of Pennsylvania, 2010.

[73] T. Gomes, C. S. oes, and L. Fernandes, "Resilient routing in optical networks using srlg-disjoint path pairs of min-sum cost," *Springer Telecommunication Systems Journal*, 2011.

[74] P. Babarczi, "Survivable optical network design with unambiguous shared risk link group failure localization," Ph.D. Thesis, 2012.

[75] Juniper Networks, "Junos 9.6 routing protocols configuration guide," 2009.

[76] Cisco Systems, "Cisco IOS XR Routing Configuration Guide, Release 3.7," 2008.

[77] Hewlett-Packard, "HP 6600 Router Series: QuickSpecs," 2008, available online: http://h18000.www1.hp.com/products/quickspecs/13811_na/13811_na.PDF.

[78] M. Menth, M. Hartmann, R. Martin, T. Čičic, and A. Kvalbein, "Loop-free alternates and not-via addresses: A proper combination for ip fast reroute?" *Computer Networks*, vol. 54, no. 8, pp. 1300–41 315, June 2010.

[79] S. S. Lor and M. Rio, "Enhancing repair coverage of loop-free alternates," in *London Communications Symposium*, London, UK, 2010.

[80] S. Bryant, C. Filfils, S. Previdi, M. Shand, and N. So, "Remote LFA FRR," Internet-Draft, Dec. 2012.

[81] Cisco Systems, "IP Routing: OSPF Configuration Guide, Cisco IOS Release 15.2S - OSPF IPv4 Remote Loop-Free Alternate IP Fast Reroute," downloaded: Apr. 2012.

[82] M. Gjoka, V. Ram, and X. Yang, "Evaluation of IP fast reroute proposals," in *IEEE Comsware*, 2007.

[83] C. Filsfils, P. Francois, M. Shand, B. Decraene, J. Uttaro, N. Leymann, and M. Horneffer, "Loop-free alternate (LFA) applicability in service provider (SP) networks," RFC 6571, June 2012.

[84] G. Rétvári, J. Tapolcai, G. Enyedi, and A. Császár, "IP Fast ReRoute: Loop Free Alternates revisited," in *INFOCOM*, 2011, pp. 2948–2956.

[85] H. T. Viet, P. Francois, Y. Deville, and O. Bonaventure, "Implementation of a traffic engineering technique that preserves IP Fast Reroute in COMET," in *Rencontres Francophones sur les Aspects Algorithmiques des Telecommunications, Algotel*, 2009.

[86] M. Menth, M. Hartmann, and D. Hock, "Routing optimization with IP Fast Reroute," Internet-Draft, July 2010.

[87] G. Rétvári, L. Csikor, J. Tapolcai, G. Enyedi, and A. Császár, "Optimizing IGP link costs for improving IP-level resilience," in *Proc. of DRCN*, Oct. 2011, pp. 62–69.

[88] L. Csikor, G. Rétvári, and J. Tapolcai, "Optimizing IGP link costs for improving IP-level resilience with loop-free alternates," *Computer Communications*, Sep 2012.

[89] C. Reichert and T. Magedanz, "Topology requirements for resilient IP networks," in *MMB*, 2004, pp. 379–388.

[90] S. Litkowski, B. Decraene, C. Filsfils, and K. Raza, "Operational management of loop free alternates," Internet-Draft, draft-litkowski-rtgwg-lfa-manageability-01, February 18, 2013.

[91] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *ACM IMC*, 2002, pp. 231–236.

[92] SNDLib, "Survivable fixed telecommunication network design library," http://sndlib.zib.de, downloaded: Apr. 2012.

[93] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *Selected Areas in Communications, IEEE Journal on*, vol. 29, no. 9, pp. 1765–1775, 2011. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6027859

[94] D. Applegate and E. Cohen, "Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs," in *ACM SIGCOMM*, 2003, pp. 313–324.

[95] L. Csikor, "GML 2 LGF converter," http://csikor.tmit.bme.hu/GML2LGF.

[96] P. Francois and O. Bonaventure, "An evaluation of IP-based fast reroute techniques," in *ACM CoNEXT*, 2005, pp. 244–245.

[97] S. Previdi, "IP Fast ReRoute technologies," in *APRICOT*, 2006.

[98] J. Tapolcai, "Reliable telecommunication networks," Hungarian Academy of Sciences, D.Sc. Dissertation, 2012.

[99] M. Nagy and G. Rétvári, "An evaluation of approximate network optimization methods for improving IP-level fast protection with loop-free alternates," in *Proc. RNDM*, Oct 2011, pp. 1–7.

[100] L. Lovász, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, vol. 13, no. 4, pp. 383–390, 1975.

[101] P. Kulaga, P. Sapiecha, and K. Sej, "Approximation Algorithm for the Argument Reduction Problem," in *Computer recognition systems: proceedings of the 4th International Conference on Computer Recognition Systems, CORES'05*. Springer Verlag, 2005, p. 243.

[102] B. Mazbic-Kulma and K. Sep, "Some approximation algorithms for minimum vertex cover in a hypergraph," in *Computer Recognition Systems 2*, ser. Advances in Soft Computing, M. Kurzynski, E. Puchala, M. Wozniak, and A. Zolnierek, Eds. Springer Berlin / Heidelberg, 2007, vol. 45, pp. 250–257.

[103] M. Nagy, J. Tapolcai, and G. Rétvári, "Optimization methods for improving IP-level fast protection for local shared risk groups with loop-free alternates," *Springer Telecommunication Systems Journal*, 2012.

[104] J. Rexford, "Route optimization in ip networks," in *Handbook of Optimization in Telecommunications*, M. Resende and P. Pardalos, Eds. Springer US, 2006, pp. 679–700. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-30165-5_24

[105] R. Teixeira and J. Rexford, "Managing routing disruptions in internet service provider networks," *Comm. Mag.*, vol. 44, no. 3, pp. 160–165, Mar. 2006. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2006.1607880

[106] M. Shand and S. Bryant, "A framework for loop-free convergence," Internet-Draft, draft-ietf-rtgwg-lf-conv-frmwk-00.txt, December 2006.

[107] M. Shand and S. Bryant, "A framework for loop-free convergence," RFC 5715, January 2010.

[108] P. Francois, M. Shand, and O. Bonaventure, "Disruption-free topology reconfiguration in ospf networks," in *IEEE INFOCOM*, Anchorage, USA, May 2007, INFOCOM 2007 Best Paper Award.

[109] F. Clad, P. Merindol, J.-J. Pansiot, P. Francois, and O. Bonaventure, "Graceful convergence in link-state ip networks: A lightweight algorithm ensuring minimal operational impact," *Networking, IEEE/ACM Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[110] 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/IFIP Network Operations and Management Symposium (NOMS)*, Apr. 2010.

[111] D. Hock, M. Hartmann, C. Schwartz, and M. Menth, "Effectiveness of link cost optimization for ip rerouting and ip fast reroute," in *Proceedings of the 15th International GI/ITG Conference on Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, ser. MMB&DFT'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 78–90. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12104-3_8

[112] D. Lucraft, A. Eremin, and F. Ajili, "Enforcing path uniqueness in internet routing," in *Proceedings of the 2006 ACM Symposium on Applied Computing*, ser. SAC '06. New York, NY, USA: ACM, 2006, pp. 399–403. [Online]. Available: http://doi.acm.org/10.1145/1141277.1141368

[113] M. Menth and M. Hartmann and D. Hock, "Routing optimization with ip fast reroute," Internet-Draft, draft-menth-ipfrr-routing-optimization-00, July 2010.

[114] H. T. Viet, P. Francois, Y. Deville, and O. Bonaventure, "Implementation of a traffic engineering technique that preserves IP Fast Reroute in COMET," in *Rencontres Francophones sur les Aspects Algorithmiques des Telecommunications, Algotel (2009)*, 2009.

[115] I. community, "Internet2 network infrastructure topology," http://www.internet2.edu/network/, 1996.

[116] B. Quoitin, "Igen," http://informatique.umons.ac.be/networks/igen/, 2005.

[117] Juniper Networks, "Example: Controlling the Cost of Individual OSPF Network Segments," Technical Documentation, Oct. 2011.

[118] W. R. Parkhurst, *Cisco OSPF Command and Configuration Handbook*. Cisco Press, 2002.

[119] Cisco Systems, "Cisco IOS IP routing: OSPF command reference," Apr. 2011.

[120] Cisco Systems, "Intermediate System-to-Intermediate System Protocol," White Paper, 2006.

[121] Juniper Networks, "reference-bandwidth (Protocols IS-IS)," Technical Documentation, Jan. 2014.

[122] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Comm. Mag.*, vol. 40, no. 10, pp. 118–124, 2002.

[123] G. Swallow, S. Bryant, and L. Andersson, "Avoiding equal cost multipath treatment in MPLS networks," RFC 4928, Jun 2007.

[124] M. Thorup and M. Roughan, "Avoiding ties in shortest path first routing," 2001, aT&T, Shannon Laboratory, Florham Park, NJ, Technical Report, http://www.research.att.com/~mthorup/PAPERS/ties_ospf.ps.

[125] C. Reichert and T. Magedanz, "Topology requirements for resilient IP networks," in *MMB*, 2004, pp. 379–388.

[126] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, New Jersey, 1993.

[127] G. Dahl and M. Stoer, "A cutting plane algorithm for multicommodity survivable network design problems," *INFORMS Journal on Computing*, vol. 10, pp. 1–11, 1996.

[128] M. Pióro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[129] P. Sarkar, H. Gredler, S. Hegde, and H. Raghuveer, "Node protecting R-LFA and manageability," Internet-Draft, draft-psarkar-rtgwg-rlfa-node-protection-01, July 8, 2013.

[130] S. Bryant, C. Filfils, S. Previdi, M. Shand, and N. So, "Remote LFA FRR," Internet-Draft, May. 2014.

[131] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, 2nd ed. Elsevier Science, 2004.

[132] Cisco Systems, "Cisco asr 901 series aggregation services router software configuration guide," http://www.cisco.com/c/en/us/td/docs/wireless/asr_901/Configuration/Guide/b_asr901-scg/b_asr901-scg_chapter_0100111.html, pp. 737–774, November 10, 2011.

[133] Juniper Networks, "Example: Configuring remote lfa over ldp tunnels in is-is networks," https://www.juniper.net/documentation/en_US/junos14.2/topics/topic-map/isis-remote-lfa-for-ldp.html, October 30, 2014.

List of Tables

List of Figures