Improving the Convergence of IP Routing Protocols

Pierre Francois

Thesis submitted in partial fulfillment of the requirements for the Degree of Doctor in Applied Sciences

October 30, 2007

Département d'Ingénierie Informatique
Faculté des Sciences Appliquées
Université catholique de Louvain
Louvain-la-Neuve
Belgium

Preamble

The IP protocol suite has been initially designed to provide best effort reachability among the nodes of a network or an inter-network. The goal was to design a set of routing solutions that would allow routers to automatically provide end-to-end connectivity among hosts. Also, the solution was meant to recover the connectivity upon the failure of one or multiple devices supporting the service, without the need of manual, slow, and error-prone reconfigurations. In other words, the requirement was to have an Internet that "converges" on its own.

Along with the "Internet Boom", network availability expectations increased, as e-business emerged and companies started to associate loss of Internet connectivity with loss of customers... and money. So, Internet Service Providers (ISPs) relied on best practice rules for the design and the configuration of their networks, in order to improve their Quality of Service.

The same routing suite is now used by Internet Service Providers that have to cope with more and more stringent Service Level Agreements (SLAs). These new SLAs are justified by the increasing use of IP networks to transport voice, video, TV broadcast, and Real-Time online Business traffic accross their networks.

Such SLAs generally define upper bounds on the packet loss ratio and on the duration of losses of connectivity. To ensure that these constraints are respected, and reach the five 9's network availability target, resilient IP technologies are required.

The goal of this thesis is to complement the IP routing suite so as to improve its resiliency.

The Internet is organized as a set of interconnected Autonomous Systems (AS), which are managed by autonomous domains such as organizations and companies. The technology used to establish connectivity inside an AS (intra-domain routing) is not the same as the one used to provide inter-domain connectivity (inter-domain routing). Hence, the recovery process upon a topological change is different, and will be improved in different ways in this thesis.

Bibliographic notes

Most of the work presented in this thesis appeared in previously published conference proceedings and journals. The list of related publications is presented hereafter :

- Achieving subsecond IGP convergence in large IP networks,
  Pierre Francois, Clarence Filsfils, John Evans, Olivier Bonaventure,
  In ACM SIGCOMM Computer Communications Review, 35(3):35-44, July 2005.

- An evaluation of IP-based Fast Reroute Techniques,
  Pierre François, Olivier Bonaventure, In Proceedings of ACM CoNEXT 2005, 244-245, Toulouse, France, October 2005

- Loop-free convergence using ordered FIB updates,
  IETF Working Group Document, draft-ietf-rtgwg-ordered-fib-00,
  Pierre François, Olivier Bonaventure, Mike Shand, Stefano Previdi, Stewart Bryant, December 2006

- ISIS extensions for ordered FIB updates,
  Internet-Draft, draft-bonaventure-isis-ordered-00,
  Olivier Bonaventure, Pierre François, Mike Shand, Stefano Previdi, February 2006

- Avoiding transient loops during IGP Convergence in IP Networks, Pierre Francois, Olivier Bonaventure,
  In IEEE INFOCOM 2005, Miami, March 2005.

- Avoiding transient loops during the Convergence of Link-State routing protocols,
  Pierre Francois, Olivier Bonaventure,
  To Appear in IEEE/ACM Transactions on Networking.

- Disruption free topology reconfiguration in OSPF Networks without protocol modifications,
  Pierre Francois, Mike Shand, Olivier Bonaventure,
  In IEEE INFOCOM 2007, Anchorage, May 2007

- Achieving sub-50 milliseconds recovery upon BGP peering link failures
  Olivier Bonaventure, Clarence Filsfils, Pierre Francois
  In Proceedings of ACM CoNEXT 2005, 31-42, October 2005, Toulouse, France
  Extended Version To Appear in IEEE/ACM Transactions on Networking

- Avoiding disruptions during maintenance operations on BGP sessions
  Pierre Francois, Pierre-Alain Coste, Bruno Decraene, Olivier Bonaventure
  To appear in IEEE Transactions on Network and Service Management

Acknowledgments

I would like to thank Olivier Bonaventure for having teached me Networking the way he did when I was a master student and during my thesis. His taste for applied research and his knowledge of the real world of networking was a major driver for my research. Olivier knows in detail how routers (the real ones) work, as well as how the people operating them work. His devotion to his work was also a major source of motivation. It's been an extraordinary experience to achieve the goals of our projects in his close company. Doing research with Olivier does not mean that you report your research results to him. It actually means that you are doing research with him, which is as rare as motivating. I would also like to acknowledge his generosity, as he's always acting for the best interest of the members of his team.

I would like to thank Clarence Filsfils, Jennifer Rexford, Roch Guerin, Laurent Mathy, Guy Leduc, Philippe Delsarte, and Yves Deville for their participation in the jury of this thesis. I have been impressed by the number of comments they provided. As I retrieved their copies of my thesis after the private defense, I actually figured out that they heavily annotated the thesis with corrections, comments, questions, clarifications, personnal thoughts, and even ideas for future papers.

Clarence Filsfils is the kind of guy that any researcher would like to interact with when doing a Ph. D. thesis. I never lacked of problems to solve because Clarence always had some to share. I don't know how many times Olivier and I came back from his office with a todo list that would occupy us for months. He trustfully shared critical information about Cisco routers and not-yet-on-the-shelve features, waiting for our comments and driving our research. My thesis is not about the hottest topic in research in networking, it is about what was hot for Clarence, and what was hot for some of his customers. This was a strong motivation for doing my job. Also, Clarence has been the toughest reviewer for our ideas; when he closed a meeting with a "good job", we actually knew that we were following the right tracks. And when it was with an "it will never work", he told us why and it became much easier to cut the wrong branches. Last but not least, I would like to thank him for being our champion for the four fundings that we received from Cisco Systems.

Other people from Cisco brought a lot to this thesis. I would like to thank Jane Butler for trusting our project and funding my Ph. D. budget. I would like to thank Mike Shand and Stewart Bryant for all the discussions we had about loop avoidance mechanisms and Fast Reroute Solutions. It's been a real fun to interact with such creative folks. I owe a lot to Stefano Previdi for having been my IS-IS mentor. It's much easier to work on IS-IS when you know you can ask questions to the guy who implements it in Cisco routers. I would also like to thank these three guys for trusting me in the standardisation process of oFIB. John Evans did a great measurement job on the performances of IS-IS, and his work was the main enabler for the first chapter of the thesis. I also thank Pranav Mehta and Pradosh Mohapatra for the information they provided on Cisco's BGP implementations.

I would like to thank Cisco Systems for its trustful support. Any opinions, findings, and conclusions or recommendations expressed in this thesis are mine and those of the other contributors, and do not necessarily reflect the views of Cisco Systems. I have also been partially supported by the Walloon government under the TOTEM project.

I would like to thank Bruno Decraene and Pierre-Alain Coste for the collaboration we had on the BGP maintenance issues. Pierre-Alain did a great job on evaluating the performance of the solutions. Bruno has also been a great companion during the IETF meetings we attended.

I would like to thank the IETF folks at the RTG, IDR, IS-IS, OSPF, and at the IRTF RRG working groups for the various discussions we had during IETF meetings. I can now confirm that IETF folks are open to discuss with people from the academia. Among them, I am especially thankful to Alia Atlas, Don Fedyk, Jean-Philippe Vasseur (thanks for the dinner in Dallas ;-)), Dimitri Papadimitriou, Robert Raszuk, Dave Ward, Alex Zinin, . . .

I would like to thank Xiaowei Yang, Ang Li, and Xin Liu for their hospitality when I visited their team at UC Irvine. I thank Amund Kvalbein and Tarik Cicic for the interesting discussions we had about IP Fast Reroute, and for their hospitality when I visited their Lab in Oslo.

I owe a big thank you to all the past and current members of my team. I enjoyed the numerous discussions I had on BGP, MPLS, and about routing in general with Cristel Pelsser, Bruno Quoitin, Sébastien Tandel, Steve Uhlig, and Virginie Van den Schrieck. I also had interesting interactions with Xavier Brouckaert, Cédric Delaunois, Sébastien Barré, Damien Leroy, Luigi Iannone, Benoit Donnet, and Stefano Iasi. All of them have been valuable for my work, and I also had a lot of fun with them at work and when we partied.

A big thank you to Geoffroy Jennes, Frédéric Grignet, Benoit Fondeviolle, Nicolas Simar, Thomas Telkamp, Jan Torrele, Ted Seely, and Michael Fiumano for their help in collecting ISP topologies. This list might be non exhaustive and I apologize to those I could have forgotten, as they were among the most precious sources of data for this work.

I would like to thank my parents, Rolande Plainchamp and Jean Francois, for having supported two stressed kids doing their masters and their Ph. D. thesis. They did everything they could to give us more comfort than needed, and always gave us a complete freedom. My sister Aurore, and Frédéric, brought a lot to me, and I am thankful for their support. I would like to thank Joseph Francois for having given me the passion of computer science.

Isabelle, I owe you an apology for the chaotic and nightly schedules, the postponed/cancelled holidays, the bursts of deadlines, and the “yes but I absolutely have to do this now, you know what it is to be a Ph. D Student”. I hope you don’t regret having given me the taste for facing research challenges when you were this damn tough teaching assistant. I would like to thank you for giving me so many advices and for pulling me back to reality in cases of emergency. Above all, I would just like to thank you for being close to me.

Contents

List of Figures

List of Tables

Part I

Background and Road Map of the thesis

Chapter 1

Introduction

In this chapter, we brielfy review the main concepts of intradomain and interdomain routing, and their dedicated protocols used on today's Internet. Readers who are familiar with intra-domain an inter-domain routing can skip this first section. Next, we introduce the key characteristics of the convergence of both intra-domain and inter-domain routing protocols in the case of topology changes.

1.1 IP Routing

The Internet Protocol (IP) is a connectionless datagram protocol used to convey data from one host location to another in a network or inter-network. Host locations are identified with an IP address. IP Routing can be seen as the set of protocols and state machines, implemented in dedicated hardware called "routers", that are used over a network or an inter-network to rule the "forwarding" of IP packets. Forwarding is the action performed by routers, which consists in transmitting an IP packet that is received on an incoming interface to another router via an outgoing interface. Routing is supposed to let routers forward IP packets in a consistent fashion so that IP packets emitted by a source actually reach their destination.

Routing in the Internet is architectured around two main components, intradomain and interdomain routing. Intradomain routing protocols or interior gateway protocols (IGP) rule the forwarding of packets within the network of a company, an organization, or an Internet Service Provider (ISP). An Interdomain routing protocol rules the forwarding of packets among a set of interconnected networks or Autonomous Systems (AS). There is only one interdomain routing protocol used in the Internet, the Border Gateway Protocol (BGP) [RLH06].

Intra- and Inter-domain routing are not performed by using the same routing mechanisms for multiple reasons. First, the objectives are not the same. While intra-domain roughly tries to achieve shortest path routing, inter-domain routing tries to achieve best commercial routing. Second, the knowledge of the topology is different. Disseminating the whole topology within an ISP network is not a problem, while ISPs are reluctant to disclose all the properties of their internal topologies to their peers and concurrents. Finally, the size of the problem are different. While the IGP routing mechanisms generally deal with a few tens to a few thousands of destination prefixes, a BGP router currently has to cope with up to 250.000 prefixes [APN], not counting the Virtual Private Network (VPN) routes that are also maintained with BGP in ISP networks.

1.1.1 An IP router

An IP router can be considered as the combination of two main elements. The "control plane" drives the decision of a router about where to forward an IP packet that reaches a router. The control plane of a router feeds a Routing Information Base (RIB). Based on this information, a process feeds the "forwarding plane" of the router, made of a table or tables downloaded to dedicated hardware, used to perform an efficient forwarding.

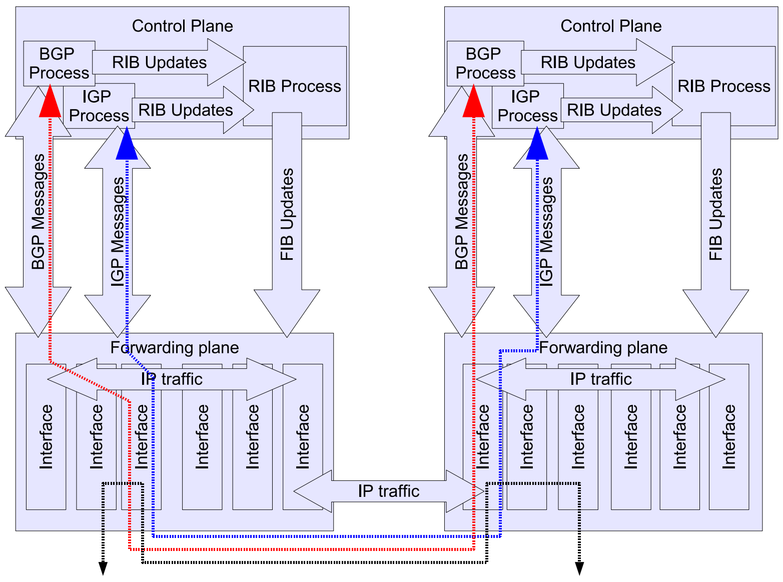


Figure 1.1: Two routers

Figure 1.1 gives a simplified view of two routers, whose control plane processes exchange routing information to build forwarding states. The control plane of a router executes routing protocol processes, e.g. a BGP process and an IGP process. Based on the routing messages that it receives from its peers, via its forwarding plane, a router maintains a Routing Information Base. A process managing the RIB is in charge of reflecting its changes to the Forwarding Information Base (FIB) which is replicated in the linecards hosting the interfaces of the router. When a data packet reaches an interface of the router, the linecard hosting the interface performs a lookup for the destination address of the packet in its locally stored

FIB, and forwards the packet to an outgoing interface corresponding to the result of this lookup.

1.1.2 Intradomain Routing

Intra domain routing consists in the set of operations performed by the control plane to provide end-to-end connectivity within a set of devices such as lines and routers owned or leased by a given organization or company, often referred to as a routing domain or Autonomous System.

It allows routers to "find their way towards" an exit point for packets destined to hosts that are beyond the borders of the network. Note that finding the appropriate exit point for a packet is a task supported by the inter-domain routing protocol.

In this thesis we will focus on "Link-State" intradomain routing protocols such as OSPF [Moy91] and IS-IS [ISO02]. We made this choice simply because these protocols are widely used in the Internet at the time of this thesis. Such protocols are referred to as Interior Gateway Protocols (IGP). Other IGP were used at the early days of the Internet, such as Cisco's proprietary EIGRP [RWS00] and RIP [Hed88]. Now, such protocols are mostly deployed in enterprise networks, but were abandoned by ISPs.

According to link-state routing protocols, each router is configured to

- Establish and maintain adjacencies with direct neighbors

  "Neighbors" are supposed to be able to reach each other at the link-layer.

- Distribute information about the state of these adjacencies through the network

  That is, a router constantly checks its ability to reach its configured neighbors using Hello messages [Moy91, ISO02] or independent failure detection mechanisms such as BFD [KW03]. This ability is summarized in a Link-State Packet (IS-IS), or Link-State advertisement (OSPF). Such information is reliably spread through the network, using flooding in IS-IS and OSPF[1]. Routers also distribute in those advertisements reachability information (IP prefixes) assigned to locally connected Local Area Networks (LANs).

- Maintain the information flooded by other routers of the network in a database, called Link-State Database (LSDB)

  Based on this flooded information, routers (nodes) build a complete representation of the network as a weighted directed graph.

- Compute shortest paths across the network

  Based on the content of the LSDB, each router computes the best paths from itself towards all the other nodes of the network. In order to translate the

[1]In this thesis, we will consider IS-IS as the reference IGP. Unless specified, what is said about IS-IS also applies to OSPF

operator requirements into shortest paths, the operator of an IS-IS network configures the metrics of the links adjacent to each router. Under the graph terminology, an operator configures the weight of each edge of the graph. Delay, bandwidth, or more complex Traffic Engineering (TE) techniques taking both components into account are the inputs for the setting of such metrics. The goal is to set metrics in order to have the shortest possible end-to-end delays, while avoiding link saturation, w.r.t. the current end-to-end traffic matrix [FRT02]. The shortest paths from a node R are path obtained by applying the well-known Dijkstra Shortest Path First algorithm [Dij59] on the network topology, using R as the root.

- Build a Forwarding Information Base (FIB)

  For a router R, the first hop on the shortest path towards a destination node D is the neighbor to which a packet destined to D must be forwarded by R. So, based on the computed shortest paths, the router can build the basic information required to perform forwarding. This information is summarized and downloaded to the linecards of the router under the form of a datastructure optimizing the lookup efficiency [Var04].

When the network is stable, the computation of the shortest paths is consistent among the routers, so that packets reach their destination or their exit point in the network by being forwarded hop by hop by intermediate routers.

Let us illustrate how the IGP works based on the Abilene network depicted in figure 1.2. Each node will establish an adjacency, using a simple Hello mechanism, with its neighbours. For example, KC will maintain adjacencies with IP, HS, and DN. Each node floods a summary of its adjacencies through the network in a link-state packet, by associating a configured metric (or weight) with each of its adjacencies.

The flooding mechanism is reliable and lets each node builds a weighted graph representing the network based on the latest link-state packet flooded by each node of the network (see figure 1.2), and stored in its Link-State Database.

Each node computes the shortest paths from itself to all the nodes in this graph. From this set of paths, the Forwarding Information Base can be computed. In our example, IP will build a FIB which reflects that KC, DN, SV, ST, LA and HS must be reached via its attached link $IP \rightarrow KC$, AT must be reached via its attached link $IP \rightarrow AT$, and CH, NY, and WA must be reached via its attached link $IP \rightarrow CH$.

All the nodes build such a FIB, so that when these are up to date, the forwarding of packets is consistent and lets each packet follow the shortest path to its destination, w.r.t. the metrics and the status of each link of the network. Note that a specific metric value, called MAX_METRIC can be configured on a link. This prevents nodes in the network from considering the link when they compute their Shortest Path Tree.

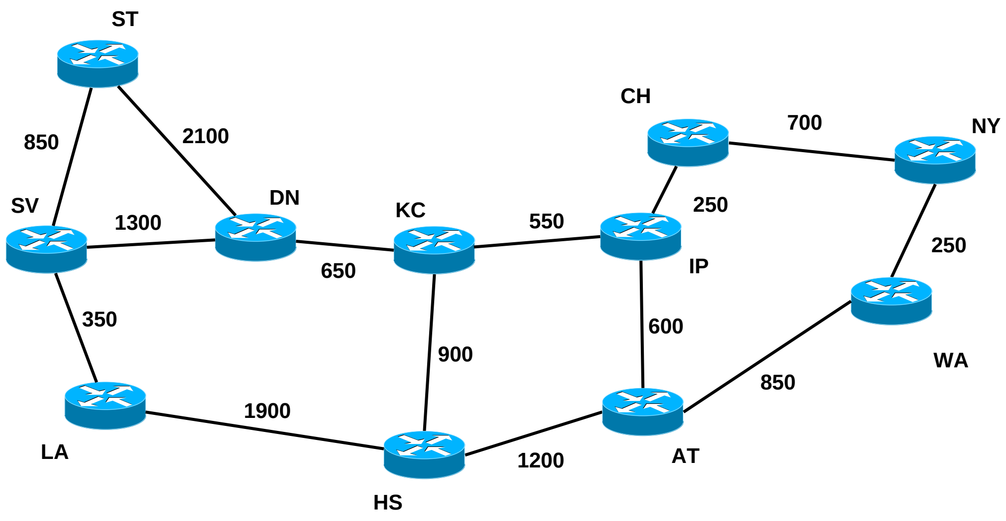


Figure 1.2: The Abilene network

1.1.3 Interdomain Routing

Interdomain routing rules how packets are forwarded between Autonomous Systems to reach their destinations. Above all, BGP aims at providing reachability throughout the Internet while respecting the business relationships and policies of all the ASes of the Internet.

BGP is a path vector protocol. That is, each BGP node propagates to its peers, for each destination (an IP Prefix) originated by an ISP, the path that it selected as best to reach the destination prefix. Each node selects a best path towards a prefix based on the set of paths that it received, and a set of policies and rules that translate its Business relationships with the neighboring networks. Among other attributes, the BGP AS-path of a route is the sequence of ASes that a packet would cross if the route is used.

Figure 1.3 illustrates an internetwork where each cloud represents an Autonomous System.

The functions of interdomain routing are split into two main features [RLH06].

Firstly, neighboring ASes exchange reachability information (BGP routes) by establishing sessions between their directly connected Autonomous System Border Routers (ASBRs). Such sessions are referred to as external BGP (eBGP) sessions. In figure 1.3, eBGP sessions are depicted as plain edges between nodes at the borders of neighbouring ASes.

Secondly, BGP paths received at the borders of an AS must be propagated through the AS. This is performed by establishing a set of BGP sessions between the routers of the AS, referred to as internal BGP (iBGP) sessions. iBGP peers are not necessarily directly connected routers, so that the IGP must provide end-to-end connectivity in the AS to support the transmission of BGP messages. There are multiple ways to establish this iBGP topology [RLH06, BC96, TMS01], we will

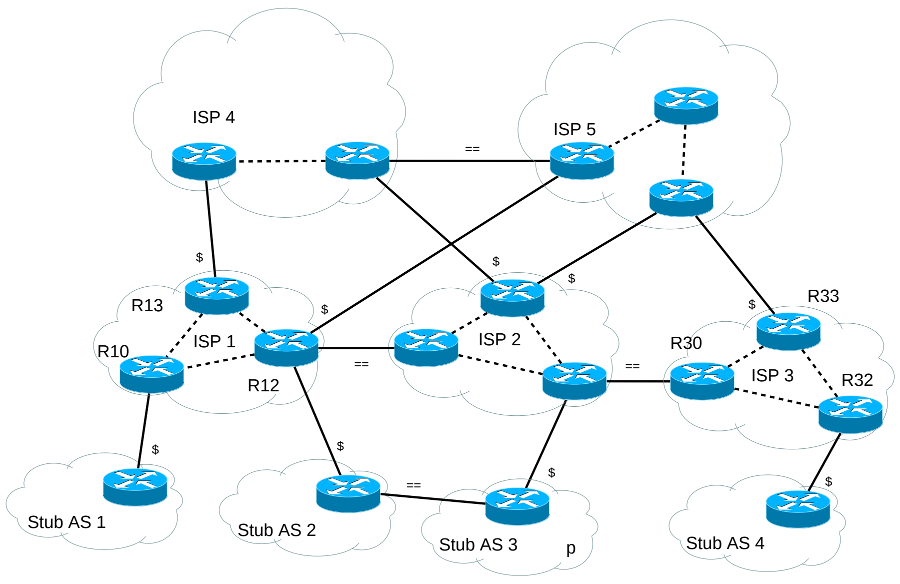


Figure 1.3: An Internetwork

discuss these specificities in chapter 7. In our illustration, the iBGP sessions are depicted as dashed edges.

Roughly, there are two types of peerings between neighboring ASes [GR00]. Each reflects a typical business relationship between the AS.

One type of peering is called Customer-Provider peering. In this case, the customer buys connectivity towards the rest of the Internet from this Provider. This type of relationship is depicted in figure 1.3 by plain edges marked with the "$" symbol. For example, "Stub AS 1" buys connectivity from its provider "ISP 1". A "stub" AS is an AS that does not provide an Internet Service to other ASes. In other words, stubs are "leaves" in the Internet topology. When the customer is itself a transit AS, i.e., when this customer is a provider of some other ASes, it will advertise reachability information about its own customers to its provider. The provider may select this peering as the exit point for IP packets destined to some customers of this customer.

Another type of peering is referred to as Shared-Cost peering. Such a type of peering is typically established between two ASes that find it convenient from a financial perspective to afford the cost of a direct connectivity using a private interconnection or an interconnection at a public Internet Exchange Point. Such peerings are depicted by using the "==" symbol in our illustration. If the customers of two ASes exchange a lot of data, it can be interesting for these not to use their (costly) providers to transit the traffic, and afford the price of a direct connectivity. Such peerings are usually established between ASes of equivalent size.

There is no "free lunch" on the Internet [GR00]. That is, an AS will not provide a transit service between neighboring ASes without a return on investment. Precisely, an AS will only propagate a BGP path selected as best to a provider or a shared cost peer if this path is via a customer. For example, routers of “Stub AS 2” will not propagate paths towards destinations within “Stub AS 3”, to “ISP 1”. Indeed, such paths are not interesting from a financial perspective for “Stub AS 2”. The feasible paths in our illustration thus contain at most one edge marked with a “==”.

To allow a translation of business relationships in control plane decisions, the BGP routing suite features import filters, export filters, and a set of rules for the selection of a best path towards a destination prefix p. An illustration of a BGP process is provided in figure 1.4.

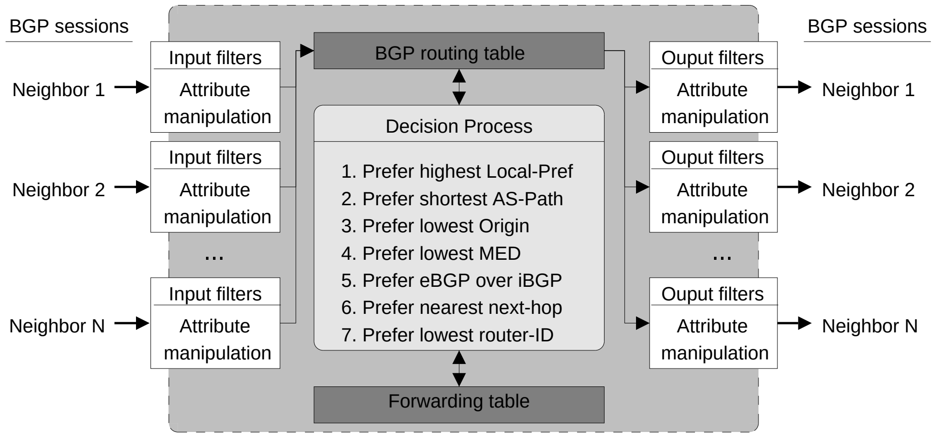


Figure 1.4: The decision process of a BGP router

Import filters

Import filters can be configured for each iBGP and eBGP sessions established by a “BGP speaker”.

Import filters can be used to ignore paths received from some BGP peers. For example, a path received from an external peer which contains the AS number of the current AS in its AS path must be removed from consideration in order to avoid routing loops. In an iBGP hierarchy, a similar rule is applied to avoid routing loops among iBGP clusters.

Import filters can also be used to filter out paths that contain an AS that is not trusted by the current AS.

Import filters are also used to tag path received from an eBGP peer with a community value [QB02] that describes the type of business relationship associated with this eBGP peer.

Those filters can also be set up to perform some operations based on the community values that have been attached to the received routes.

Decision process rules

The BGP decision process is made of a set of rules that dictate the selection of the best path towards an IP prefix, among the set of paths were not dropped by the import filters configured on the current node. These rules are applied in their order of presentation. At each step of the process, some paths will be removed from consideration, and the process terminates when one single path to p remains.

1. Ignore paths with unreachable next hops

   This rule is aimed at preventing the current node from selecting a path via a given peering link if the underlying IGP does not consider the peering link to be reachable from the current node. This prevents the BGP decision process from selecting a path that is not usable.

2. Prefer paths with the highest Local-Pref

   The local pref is an attribute of a path that is set by a router that receives the path from a neighboring AS. Typically, paths received from a customer will be set with a higher local pref than paths received from a shared cost peer, which have themselves higher local-pref values than the paths received from a provider. This attribute translates the preference of an exit point in the local Autonomous System, for a given destination IP prefix.

3. Prefer paths with the shortest AS-PATH

   The AS-PATH of a path represents the sequence of ASes that a packet destined to the prefix will follow, if the router selects that path. The length of this path can be artificially increased by routers [QUP+03], as these can manipulate this attribute by for example adding the number of the current AS multiple times.

4. Prefer paths with the lowest Multi Exit Discriminator (MED)

   The MED is an attribute of a path that is set by an ASBR when it propagates the path to an ASBR of a neighboring AS. Its goal is to drive the selection of the peering link over which packets will be forwarded, in order to perform incoming traffic engineering. Typically, a customer will be allowed to set different MED values for the paths to a prefix that are propagated over a set of redundant eBGP peering links with the provider. The idea is that providers are paid to carry the bits of their customers, so that these are given a mean to select the incoming point for packets to a given destination. There are multiple means of applying this rule, as explained in chapter 8.

5. Prefer paths received over an eBGP session over ones received over an iBGP session

   When multiple paths remain after the application of the previous rules, a router will prefer paths which would let packets directly go out of the current

AS via one of its external link. This is actually the first step of the "hot potato" rule [TSGR04].

6. Prefer paths with the nearest BGP next-hop (hot potato)

   The remaining paths are all equivalent from a business perspective. It is thus recommended for a router to select the one(s) that would consume the less ressources in the current AS. When shortest path routing is applied in the IGP of the network, this turns to select the path whose exit point (the BGP next-hop) is the nearest from the current router, in terms of the IGP distance computed by the IGP process of the router.

7. Apply an arbitrary, stable, rule

   From this stage, the tie-breaking is roughly an implementation choice, which is aimed at selecting a path and ensuring the stability of this selection when the set of paths received by the router does not change anything on the application of the previous rules. Some implementations let the router select the paths that has been present in the set of available paths for the longest time. Some implementation select the path based on a numerical ordering of the IP address of the nexthop [RLH06].

Export filters

Export filters are configured for each iBGP and eBGP sessions established by a BGP speaker.

Based on the community attached by the ASBR which describes the business type of a path, an ASBR will be able to decide wether the path selected as best in the decision process can be propagated to some other eBGP peers. These filters will typically prevent an AS from serving as a transit AS between two providers or two shared cost peers or between a provider and a shared cost peer.

These filters can also be used to set MED values, or tag the routes advertised to a peer with some communities to drive the decision process of the routers that will process the propagated route.

1.2 Routing Convergence

Routing convergence is the transition from a routing and forwarding state to another state. Convergence is triggered by a change in the topology of the network or inter-network. Such changes can be caused by two main types of events.

First, events can be **sudden** physical failures of a lower layer transmission media such as fiber cuts and power failures. Note that when the transmission media comes back into service, the corresponding event can also be considered as sudden, even though it is a "good" event for the network.

Second, events can be manual reconfiguration of the network and maintenance operations that affect the routing and forwarding process. These are justified by software and hardware upgrades, removal, or installation of a link or a node of the network. These events are **predictable** as they are planned in advance by the operators. Such events are frequent in ISP networks [MIB$^+$04].

Sometimes, a planned maintenance of hardware can be perceived as sudden by an AS. For example, a maintenance performed on a leased line or in an Internet eXchange Point can be seen as an urgent event by the routers that are using these ressources.

1.2.1 Intra-domain convergence

Inside a domain, a routing convergence will take the form of a modification of the Link-State information of the nodes that are adjacent to the affected link(s) or node(s). This information will be flooded and routers will have to recompute their shortest paths and adapt their FIB according to these new paths.

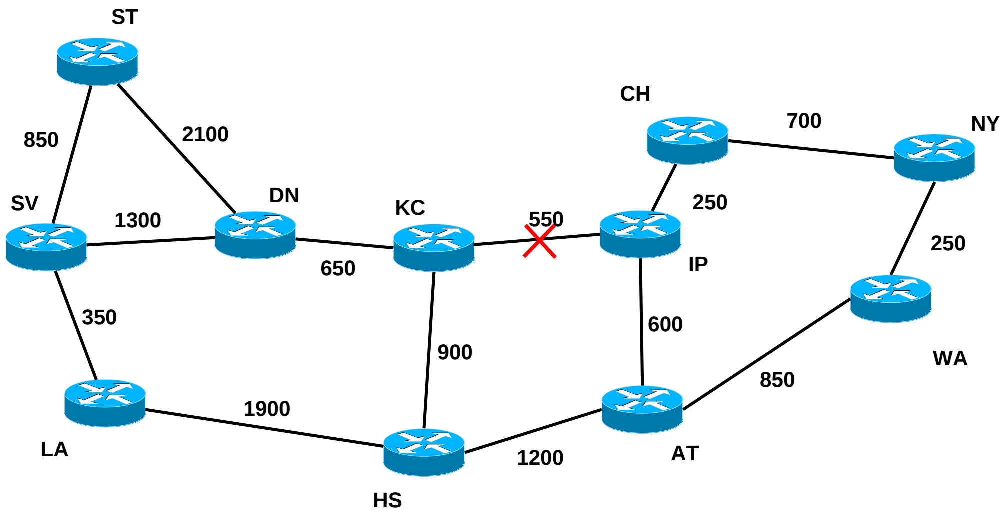


Figure 1.5: A link failure in the Abilene network

Let us illustrate an intra-domain convergence in the Abilene topology depicted in figure 1.5. Let us assume that the link between IP and KC fails. IP and KC will both trigger the generation of an update of their link state packet which respectively prune the link $IP \rightarrow KC$ and $KC \rightarrow IP$ from the topology. Upon the reception of one of these link state packets[2], a router recomputes the shortest path tree rooted on itself, and download the consequent updates of FIB entries to its forwarding engines.

[2]Due to the “two-way connectivity” check rule applied in common IGPs, the absence of a link $A \rightarrow B$ prevents the reverse link $B \rightarrow A$ from being used, whatever its current state.

IP and KC will probably be the first nodes to update their FIB, as they are the first nodes to be aware of the event. IP will update its FIB for the destinations KC, DN, SV, ST, LA and HS. All these destinations will be "rerouted" to AT. AT, NY, and WA will also perform some FIB updates to reflect the new shortest paths in the network, once they receive one of the link-state packets describing the failure. Due to the distributed nature of the convergence process, some routers can be transiently inconsistent. For example, the packets destined to DN that are rerouted by IP towards AT will loop back from AT to IP until AT also has updated its FIB entry for DN. Some distant forwarding loops can also occur, for example, between WA and NY.

The recovery upon a failure in traditional IGPs is a thus distributed process, where distant nodes are also in charge of restoring the connectivity within the network. We will see in chapter 2 that this aspect of the convergence process prevents us from ensuring a very low upper bound on the convergence time of the IGP.

1.2.2 Inter-domain convergence

In the global Internet, events triggering a convergence modify the set of paths on which adjacent routers base their best path selection. The resulting best path selection may change, so that BGP updates will be advertised to their peers. Due to its path-vector nature, the routing information in a node is incomplete so that the information about the failure is not necessarily sufficient to perform the right, definitive, update in the FIB of a router. An "exploration" is thus performed during the convergence of BGP [LABJ00].

Let us illustrate this exploration with an example based on figure 1.6. If the usual business relationships among ISPs are applied, the BGP speakers of $ISP\ 1$ initially select $ISP\ 2$ as the neighbouring AS to which packets destined an IP prefix p homed in $Stub\ AS\ 3$ must be forwarded.

Let us assume that the peering link of $R12$ in $ISP\ 1$ with router $R22$ in $ISP\ 2$ goes down. As a result, $R12$ will run its BGP decision Process to select another path for prefix p, advertised by $Stub\ AS\ 3$. It will select its external path $ISP\ 5, ISP\ 2, Stub\ AS\ 3$. $R12$ will send a "BGP path update" message towards $R10$ and $R13$ so as to reflect the change.

$R10$ will rerun its decision process and select this path as its best path for p. It will thus send a path update message over its peering link with $Stub\ AS\ 1$ to update the information relative to the path it uses to reach p.

$R13$ will also apply its decision process, but it will prefer its own external path for p via $ISP\ 4$ upon the application of the fifth rule of the decision process described in the previous section. As a result, $R13$ will itself send a path update message towards its iBGP peers.

$R10$ may change its decision about its best path for p again. Indeed, $R13$ could for example be closer than $R12$ according to the IGP metrics, so that $R10$ will once again update its routing tables and send a new path update message towards $Stub\ As\ 1$.

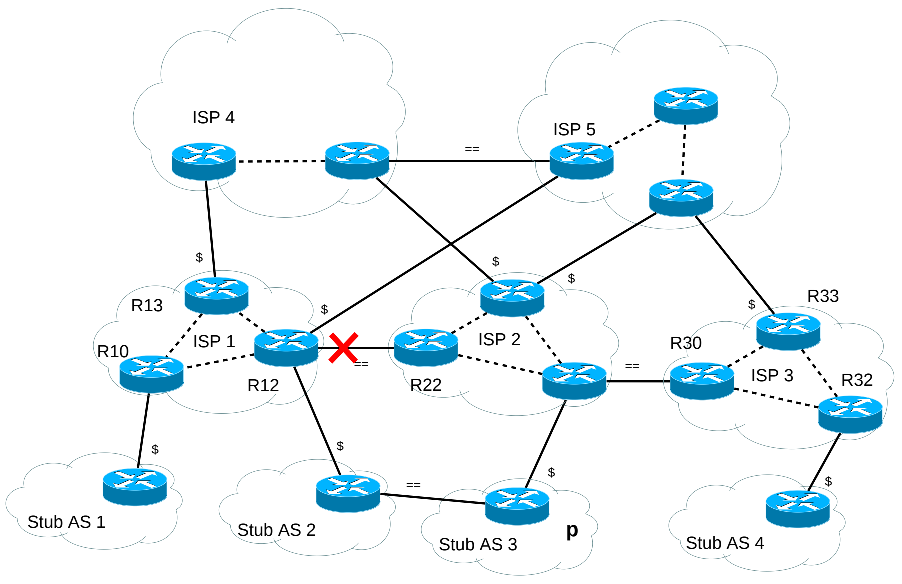


Figure 1.6: A failure in an Internetwork

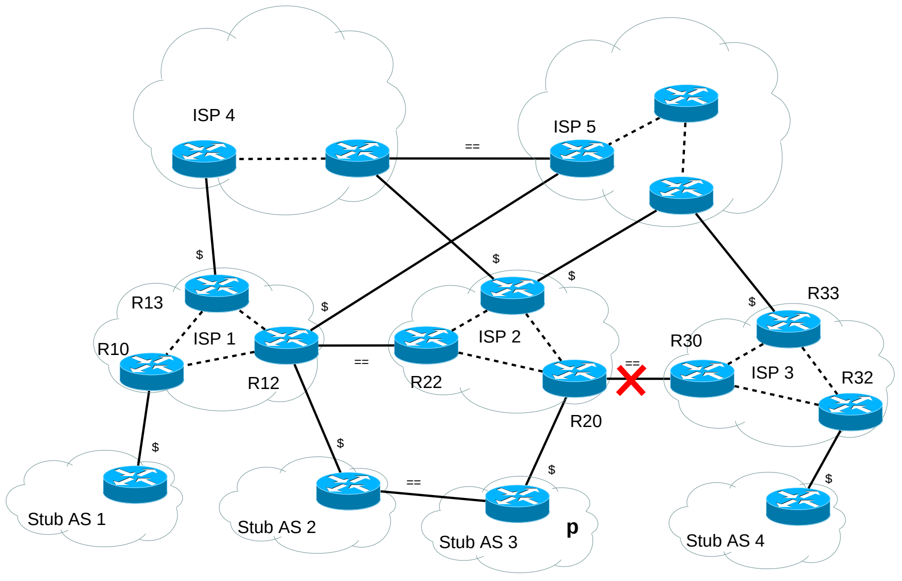


Figure 1.7: A failure in an Internetwork

Worst convergence scenarios can occur, where some routers transiently lack of a path towards a given IP prefix and drop traffic. Let us for example consider in figure 1.7 the failure of the peering link of $R30$ in ISP 3 with $R20$ in ISP 2.

At the failure time, neither $R30$ nor $R32$ know the path towards p via $ISP\ 5$. Indeed, $R33$ did not propagate this provider path as it was less prefered compared to the initial path via the failing shared cost peering link. As a result, $R30$ will start dropping packets destined to p, and $R32$ will do the same upon the reception of the BGP "withdraw" message from $R30$. $R32$ will propagate this withdraw message towards $Stub\ AS\ 4$, which will start dropping packets as well. When $R33$ processes the withdraw message, it will finally propagate the path via $ISP\ 5$ towards its iBGP peers, so that the forwarding of packets destined to p will be recovered.

These aspects of the convergence (path exploration and transient lack of alternate paths), and the number of IP prefixes potentially affected by one single event renders a BGP convergence an order of magnitude slower than a Link-State IGP convergence [WMW+06].

Thesis Road Map

Our goal is to reduce at most the packet losses that are due to the IP routing protocols and their convergence upon topological changes. To achieve our goals, we rely on a separation between the recovery process following a topological change and the transition to the new optimal forwarding state in the (inter-)network, according to this change. This approach will be followed in the second and third part of the thesis.

The second part of this thesis is dedicated to intra-domain routing.

We start our work with a simulation-based evaluation of the convergence time of Link-State Routing protocols like OSPF or IS-IS (**chapter 2**). This study is based on white-box measurements performed on high-end routers to evaluate the impact of hardware and software on the components of the convergence. Then, we injected those measurements in a simulator to evaluate the convergence time in large ISP networks. We propose configuration rules and potential modifications to the behaviour of routers to improve the convergence time. The conclusion of this part is that, by essence, the convergence of intra-domain routing protocols currently used on the Internet allows a sub-second convergence, but hardly meets a sub-50 msec recovery requirement. Thus, such protocols must be complemented by other techniques to achieve this goal, which motivates the approach of the thesis.

After that, in **chapter 3**, we analyse in detail IP Fast Reroute techniques aimed at protecting links from sudden failures. The main idea underlying such techniques is to prepare routers to **locally recover** the reachability among hosts once a device surrounding them in the topology suddenly fails. The actions performed by the routers will let packets reach their destination by following paths around the failed component, ensuring that packets will not be looped back to the node that activated the protection. For example, in our illustration of figure 1.5, IP will be prepared to perform a simple operation when it detects the failure of $IP \rightarrow KC$, so that packets will be deviated around the failed link without suffering from transient loops. Note that the resulting end-to-end paths might differ from the end-to-end paths that the routing system would provide with respect to the new topology and the configuration of the routing system. Such "Fast Reroute" techniques have been proposed for MPLS networks in [LYN$^+$04] and for IP networks in [AZ07, Atl06, BFPS05, BSP06]. We present an analysis of the coverage and the stability of such techniques based on real ISP Topologies, and propose enhancements to some of them.

In a second step, the routing system will be adapted so as to allow a packet **loopfree and lossfree transition** from the recovered state of the network provided by the "Fast Reroute" mechanisms to the post-convergence state dictated by the new topology and the configuration of the routing system. For example, the solution allows a transition of the recovered path from NY to DN upon the failure of the protected link $IP \rightarrow KC$ which prevents loops from occuring along the link $NY \leftrightarrow WA$ for packets destined to the west part of the topology. We motivate such solutions in **chapter 4** by providing an analysis of ISP topologies that illustrates the conditions under which such transient inconsistencies occur, and that gives insights on the amount of transient forwarding loops that can occur during the convergence in real ISP networks.

We will propose two different solutions to the problem.

In **chapter 5**, some modifications to the IGPs will be proposed to avoid transient forwarding loops and losses. Loop avoidance techniques were already proposed for shortest path routing system in the litterature, especially for distance vector protocols, notably in [GLA89, JM82, SCK$^+$03]. Actually, current state of the art of transient loop avoidance in link-state routing protocols were transpositions of such solutions to link-state routing protocols. But these thus missed a very nice property of link-state protocols; the nodes of such systems know the whole topology. Given a topological change, which can be explicitely identified in link-state routing systems, nodes are able to perform the final, post-convergence updates to their FIB. This property allowed us to define a much simpler loopfree ordering, which dramatically reduces the computation and signalling overhead of its implementation. At the early stages of the project[3], people from Cisco Systems came up with a rank-based implementation of such an ordering, where nodes computed a time at which they were allowed to update their FIB without introducing forwarding loops, while we came up with a fully distributed implementation of the same ordering. The solution that is depicted in **chapter 5** is a mix of both solutions, that we decided to push forward as it gives a very fast implementation of the ordering while ensuring its robustness.

In **chapter 6**, we propose another technique to avoid forwarding loops, which relies on sequences of link metric reconfigurations that permit routers to adapt to a topological change without introducing forwarding loops. An evaluation of the time required to adapt to a topological change is presented for both solutions. Its main advantage is that the solution does not require a single modification to the routing protocol itself, as reconfiguring a IGP link metric is a "feature" that exists by essence in a link-state routing protocol implementation. This solution avoids the long lasting standardization process that is required by protocol modification proposals, and it would also allow different vendors to implement different flavors, optimizations, or simplifications of the proposed solution.

The third part of this thesis is dedicated to inter-domain routing.

Fast Reroute Solution as described for intra domain routing do not apply to

[3]Improving the convergence of the Igp (ICI)

protect BGP peering links. We will propose such a feature for BGP in **chapter 7**. Basically, we will introduce a new type of BGP route that would let an ASBR discover alternate peering links that can be used to protect its own eBGP peering links. We will illustrate the problem by showing that BGP peering links fail as often as intra domain links. Then, we will present the requirements associated with the protection of peering links, such as the respect of policies. We will then present the solution in details, from the design of a FIB that supports the solution, to the messages that routers need to exchange to establish the protection. We will also discuss its applicability of the solution to different kinds of peering links, and introduce the problem of the transition to the post-convergence state after the activation of the local protection.

In **chapter 8**, we propose modifications to BGP that allows the shutdown of an inter-domain peering link without introducing transient unreachability of nodes accross the network. We also describe maintenance practices that operators can apply to avoid transient unreachabilities. Solutions relying on modifications to BGP can also be applied in the case of a sudden failures of links, provided that they are protected with the techniques proposed in chapter 7.

Part II

Improving the convergence of intra-domain routing protocols

Chapter 2

Achieving sub-second convergence in an ISP network

In this chapter, we describe and analyze in details the various factors that influence the convergence time of intra domain link state routing protocols. This convergence time reflects the time required by a network to react to the failure of a link or a router. To characterize the convergence process, we first present the results of detailed measurements performed on router platforms that are currently deployed in ISP networks. We determined the time required to perform the various operations that are required by the convergence process of a link state protocol. We then build a simulation model based on those measurements and use it to study the time required to recover intra domain end-to-end reachability after a failure in large ISP networks. Our measurements and simulations indicate that sub-second link-state IGP convergence can be conservatively met on an ISP network without any compromise on stability.

2.1 Introduction

OSPF and IS-IS are the link-state (i.e. LS) Interior Gateway Protocols (i.e. IGP) that are used in today's IP networks [Moy91, Ora90]. Those protocols were designed when IP networks were research networks carrying best-effort packets. Their initial goal was to allow the routers to automatically compute their routing and forwarding tables without consuming too much CPU time during network instabilities. This explains why, until recently, the typical LS IGP convergence in Service Provider networks used to be in tens of seconds [AJY00, Mar02].

Today, IP-based networks are used to carry all types of traffic, from the traditional best-effort Internet access to traffic with much more stringent requirements such as real-time voice or video services and Virtual Private Networks. Most network providers are currently deploying or have already deployed converged networks that enable them to offer all types of data and multimedia services over a single IP-based infrastructure.

Some of those services have strong requirements in terms of Quality of Service (QoS) and restoration time in case of failure. QoS received a lot of attention from the research community and the equipment vendors since the late nineties. Today, ISPs can deploy QoS mechanisms inside their own network to protect mission critical services [FE05]. Much tighter Service Level Agreements (SLA) are now required in terms of service restoration in case of failures, leading to LS IGP convergence requirements from sub-3-second to sub-second [ICBD04, Fil04a]. This chapter shows that sub-second LS IGP convergence can be conservatively met on a SP network without any compromise on stability[1].

The chapter is structured as follows: we firstly provide an overview of a typical IS-IS convergence. While for ease of reading we use the IS-IS terminology throughout the chapter, the analysis equally applies to OSPF. We then characterize in Section 2.3 each of the components of the convergence on a single router in terms of its rapidity of execution and robustness against unstable network conditions. Next, in Section 2.4 we build a simulation model based on those measurements and use it in Section 2.5 to evaluate the convergence time in two large SP networks and the influence of the characteristics of the network itself on the convergence.

2.2 Link-State IGP Convergence

A brief overview of IS-IS can be found in chapter 1. A more detailed description of IS-IS can be found in [Ora90, WR03]. IS-IS supports a multi-level hierarchy, but as most large ISPs running IS-IS use a single level, we do not consider it in this chapter.

The overall operation of an IS-IS router can be sketched as follows. First, the router will exchange HELLO PDUs with its neighbours to determine its local topology. The router will then describe its local topology inside a link-state packet (LSP) that will be reliably flooded throughout the network. This LSP will contain at least the identifier of each neighbour, associated with the metric of the directed link from the router to this neighbour. Note that a mechanism called two-way connectivity check allows routers to use a link (i.e. to consider it as being part of a path to a given destination) only if both adjacent routers describe it as being up and running [Ora90]. The LSP will typically also contain information about the IP addresses attached to the router as well as various optional parameters such as the Traffic Engineering information [WR03]. When broadcast networks, such as Ethernet LANs, are used, the situation is slightly different. On each broadcast network, the IS-IS routers attached to the network will elect a designated router. This router will "represent" the broadcast network and will generate a LSP describing this network and the routers attached to it. Thus, an IS-IS router attached to several broadcast networks may generate several LSPs.

[1]A part of this chapter has been published in [FFEB05]

With IS-IS, two types of events can force a router to flood a new LSP. First, a new LSP is generated and flooded each time the information contained in the LSP (neighbours, IP addresses, metrics, TE information, ...) changes. Timers can be configured to throttle the rate at which such changes are flooded through the network. Bandwidth thresholds have also been made configurable to control the rate at which TE information updates are flooded by routers [AGK99]. Second, to avoid problems in case of undetected memory or transmission errors, each LSP has a lifetime. Upon expiration of the LSP's lifetime, its parent router must flood it again. While the IS-IS specification did mention a default lifetime of 20 minutes, in practice, large SP's usually set it to its maximum value (i.e. 18 hours) to reduce the background flooding noise.

To ensure that LSPs are reliably flooded throughout the network, each LSP is acknowledged on each link. When the IS-IS specification was written, the link speeds were much lower (i.e. T1) and the CPU's were much slower. Hence, in order to prevent LSP packets from congesting links and overloading neighbours' CPU's, a pacing timer of 33ms was specified between any two consecutive LSP transmissions on the same link.

Once a LSP describing a topology change has reached a router, this router updates its Link State Database (LSDB) which triggers a request to update the routing table (i.e. commonly called Routing Information Base, RIB). To update its RIB, a router must compute is Shortest Path Tree (SPT) based on the information stored in the LSDB. The RIB update itself triggers the update of the Forwarding Information Base (FIB). The FIB is a copy of the RIB that is optimised for forwarding efficiency. On distributed platforms, the convergence process ends with the distribution of the FIB modifications to the various linecards of the router.

In summary, a typical IS-IS convergence after a link failure can be characterized as $D + O + F + SPT + RIB + DD$ where D is the link failure detection time, O is the time to originate the LSP describing the new topology after the link failure, F is the complete flooding time from the node detecting the failure (i.e. Failure node) to the rerouting nodes that must perform a FIB update to bring the network in a consistent forwarding state, SPT is the shortest-path tree computation time, RIB is the time to update the RIB and the FIB on the main CPU and DD is the time to distribute the FIB updates to the linecards in the case of a distributed router architecture.

2.3 Components of the convergence time

This section characterizes each convergence component in terms of its rapidity of execution and its robustness against unstable network conditions.

The measurements presented here have been obtained by instrumenting a Cisco 12000 router with a GRP[2] processor and Eng2 PoS linecards. To perform measure-

[2]In reality, most such types of routers are now equipped with PRP2 processors which are more than twice as performant as the GRP and with more recent linecards with much faster LC CPU's. This

ments, the unit under test was inserted in an emulated IS-IS network of 700 nodes and 2500 prefixes. It was running BGP with 160000 routes. SNMP probes were sent to the UUT to obtain an average 5-min CPU utilisation of 30% (it is rare for such routers to have an average CPU utilisation higher than 10%). On top of this excessive load, 16 BGP flaps per second were continuously sent to further stress the router.

2.3.1 Router Architecture, Processor Performance, Operating system

A distributed router architecture with hardware packet processors is very well suited for faster IS-IS convergence as it dedicates all its CPU power to the sole control plane operation: the central CPU (also called RP) handles all the routing protocol operations (IGP, BGP, RIB, FIB) and downloads the FIB update to the CPU's on the linecards which write them into the hardware packet processors.

The operating system run by the RP and LC CPU's implements a process scheduler with multiple priorities and preemption capabilities. This allows for example for the IS-IS process to be scheduled immediately upon link failure even if a process of lower priority was running at that time (e.g., a maintenance process).

During a convergence on a distributed platform, at least two processes of the same priority must share the CPU: the IS-IS process to update the RIB and FIB, and the so-called IPC process to distribute the resulting FIB modifications to the LC CPU's. The RIB update being the key bottleneck, prioritization techniques have been developed to ensure that IS-IS starts the RIB update with the most important prefixes. To ensure that these most important modifications are immediately distributed to the linecards, a small process quantum is often used (i.e. $50ms$). In practice, this leads to the following operation: immediately after completing SPF, IS-IS starts updating the RIB with the most important prefixes. When the $50ms$ quantum is over, the IPC process is scheduled and these most important updates are distributed to the linecards. When the $50ms$ quantum is over, the IS-IS process is rescheduled and the RIB updates continues followed by the IPC distribution and so forth. In the worst-case, in very large networks with lots of IS-IS prefixes, ten or more such rounds may be required which would lead to worst-case convergence time for the last prefix over the second. The use of this RIB update prioritization technique and the parallelism between the RIB update and the FIB distribution to the linecards ensure that the most important prefixes are updated well under a second as we will see later.

2.3.2 Link Failure Detection

The dominant use of Packet over SDH/SONET (POS) links in SP backbones and hence the ability to detect a link failure in a few tens of milliseconds is a major enabler of sub-second IGP convergence.

slower hardware combination was chosen to emphasise the conservative property of the analysis.

Inbuilt mechanisms in SDH and SONET (Loss of Signal (LOS), Loss of Frame (LOF), Alarm Indication Signal (AIS) etc.) allow the linecard hardware to detect the failure in less than 10 milliseconds. Immediately thereafter, a high-priority interrupt is asserted on the LC CPU which causes a POS routine to be executed. This routine enforces a user-specified hold-time which, if configured, delays the communication of the failure to the central CPU to allow for the SONET/SDH protection to occur (sub-$50ms$ in most cases, sub-$110ms$ in extreme conditions)[VPD04]. If such protection is not provided by the SONET/SDH network, then the user will not configure any such delay and the failure will immediately be signalled to the common CPU [Cisc]. This latter will update the interface status and hence schedule IS-IS for reaction. We repeated 5000 POS failures and measured the delta time between the high-priority interrupt on the LC CPU and when the IS-IS process is scheduled on the main CPU.

The objective of the test is to characterize, in a loaded distributed-architecture router, how much time is added by the software infrastructure to the sub-$10ms$ failure detection provided by the SONET hardware. In the lab, the measured Percentile-90 was $8ms$ (for a total worst-case detection of $10 + 8 = 18ms$). The measured percentile-95 was $24ms$ and the worst-case measurement was $51ms$.

This confirmed the theoretical expectation: in almost all the cases, the rapid SDH/SONET hardware detection on the LC is complemented with a prompt signalling from the LC CPU to the main CPU leading to an overall detection of less than $20ms$. When the control plane load increases, although very rare as confirmed in our test results, it becomes possible that another process was owning the CPU when the detection occurred on the LC. In the worst-case, this process is, first, of the same priority as IS-IS (i.e. BGP); second, it was scheduled just before the failure occurred; third, it is busy enough to consume its full quantum of $50ms$.

While POS represents the vast majority of link types between routers, the same sub-$20ms$ property was confirmed for two other common link types: back-to-back Gigabit Ethernet and Spatial Reuse Protocol (SRP).

When SONET/SDH link or path alarms are cleared (indicating a link or node recovery), timers are used to hold the interface down for an additional 10s before the routing protocols are informed to ensure robustness against unstable situations such as flapping links. Router vendors have generalised the interface state dampening concepts to non-POS links and have extended it with adaptive timers, which can change their rate of responsiveness based upon the stability of the interface [Cisa]. This "dampening" of good news protects the routing protocol from network instability, caused by flapping links for example.

For link-layers which do not have such a link management capability, the worst-case time to detect a failed neighbour is dependent upon the hello mechanism of the IGP. With the use of faster IGP hellos [PMA01], the worst-case time to detect neighbour failure can be much reduced, resulting in improved convergence times. The IGP Hello protocol has been however built mainly for adjacency discovery and parameter negotiation and is most often supported on the central processor card of a distributed-architecture router. It is thus unlikely that very fast failure detection may be achieved, as it would require an intensive use of the central processor.

To implement faster hello's, Bidirectional Forwarding Detection (BFD) can be used [KW06]. The main advantage of BFD over the Hello messages of IS-IS is that BFD can be easily implemented on the linecards themselves. Thus, shorter time limits can be set and a fast detection is possible without major impact on the main CPU.

In conclusion, the majority of the routers benefit from such very fast failure detection (sub-$20ms$) mechanisms without any compromise on stability.

2.3.3 LSP Origination

A rapid dissemination of updated Link State Packets is essential for rapid convergence. But an unstable device can lead to the generation of an excessive number of LSPs.

Traditionally LSP generation timers have been statically defined, that is they were set to fixed values [Ora90]. Those statically defined timers have been set to limit the routing protocol overheads incurred during times of network instability, more precisely when links flap. This consequently also impacts the convergence times that can be achieved in a stable network.

To overcome this problem and to achieve both rapid and stable convergence, dynamic, rather than static, timers have been introduced to control the LSP generation process [Mar02]. The concept of dynamic timers is that they can adapt their duration and hence responsiveness depending upon the stability of the network. When the network is stable, the timer is short and ISIS reacts within a few milliseconds to any network topology changes. In times of network instability, however, the timer exponentially increases in order to throttle the rate of IS-IS response to network events. This scheme ensures fast exchange of routing information when the network is stable (down to a few ms to 10's of ms) and moderate routing protocol overhead when the network is unstable, thus allowing the network to settle down.

The duration between when ISIS is scheduled and the LSP generation is finished was measured on the previously described testbed: the measured percentile-50 and -100 were $8ms$ and $12ms$.

In conclusion, the origination time is extremely fast ($<= 12ms$) without any compromise on stability.

2.3.4 Flooding

The flooding time from the Failure node to the Rerouting nodes is the sum at each hop of the bufferisation, serialization, propagation and the ISIS processing time.

Serialization, the time taken to clock the packet on the link, is negligible on a SP backbone (1500 bytes are sent in less than 5μs at OC48 speed). Bufferization is also negligible: most SP networks are capacity planned outside congestion [Cas01] and routers prioritize routing updates through input and output buffers, as proposed notably in [SVKD00].

We will evaluate the impact of the propagation delay on the IGP convergence with the simulation model in section 2.4. We focus the remainder of this section on optimizations for fast flooding time per hop [Cisb] and their lab characterisation.

First, a single-threaded IS-IS implementation must ensure that the LSP is flooded before the RIB is updated. Indeed, this latter can take several hundreds of millisecond and such a delay would jeopardize the overall network convergence when the local node is not the sole rerouting node.

A second important optimization enabled with fast flooding behaviour is related to the pacing timer. The value of $33ms$ suggested by the IS-IS specification [Ora90] is outdated by current link speeds (40G nowadays vs T1 15 years ago) and processor performance. Using this timing is potentially quite damaging to the IS-IS convergence time. Indeed, upon a node failure, in the worst-case, all the LSP's of the neighbours of the failed node are required to compute the correct alternate path(s). Assuming a node with 10 neighbours, we see that with the default pacing timer suggested by the IS-IS specification, the last LSP could be unnecessarily delayed by $300ms$.

Fast flooding has been introduced to overcome the effects of pacing on convergence. Its ideal implementation bypasses pacing on LSPs that describe a new link-state change event, and applies pacing on Refresh and TE LSPs. Such an implementation requires that link flaps do not trigger bursts of LSP origination describing unstable link states. More conservative implementations of Fast Flooding let routers bypass the pacing on the same kinds of LSPs, but the burst size is controlled and pacing is re-applied by routers detecting that a configurable amount of LSPs have been fast flooded within a configurable amount of time [Cisb].

In order to characterize the resulting fast-flooding behaviour, we send a LSP to the previously described UUT and measure the time until the same LSP is seen on its other interfaces. The measured Percentile 90, 95 and 100 for 1000 measurements were respectively $2ms$, $28ms$ and $52ms$. As for the link failure detection, this worst-case is measured very rarely as it requires the combination of two conditions: a process of the same priority as IS-IS was scheduled just before the event and was busy enough to consume its entire process quantum. In practice, the probability of occurrence will even be smaller and this worst-case should be neglected. Indeed, due to the meshing of the networks, several parallel paths exist between the failure and rerouting nodes and hence for the worst case to really occur, the conditions must occur at the same time along all the parallel paths.

In conlusion, we have shown that the time to flood one LSP is negligible compared to the sub-second convergence objective.

2.3.5 SPT Computation

The dynamic timers described in the context of controlling LSP generation in section 2.3.3 have also been applied to control the occurrence of SPF recalculations

[Cis04b]. This allows IGPs to be tuned such that when the network is stable, their timers will be short and they will react within a few milliseconds to any network topology change. In times of network instability, however, the SPF timers will increase in order to throttle the rate of response to network events. This scheme ensures fast convergence when the network is stable and moderate routing protocol processing overhead when the network is unstable.

The computational complexity of a typical implementation of the SPF algorithm is $O(nlog(n))$ where n is the number of nodes [Dij59]. Therefore, in a network designed for fast IGP convergence it is best practise to minimise the number of nodes in the topology. For example, Ethernet connections used as point-to-point links between routers should be modelled by the IGP as point-to-point links rather than multi-access links to avoid introducing too many pseudo-nodes in the topology.

Incremental SPF (iSPF) [MRR79] is an important algorithmic optimization to SPF computation and hence should be considered for a faster IGP convergence [AJY00]. iSPF analyses the impact of the new LSP/LSA on the previously computed SPT and minimises the amount of computation required. For example, if the change only involves "leaf" information, e.g., a new IP prefix has been added to node X, then the previous SPT is still correct and all that is required is to read the best path to node X and add an entry in the routing table for the prefix via that path. This operation is called partial route calculation and is notably described in [Cal90]. Another straightforward example relates to link deletion. When the topological change does belong to the previous SPT, iSPF determines the subset of nodes impacted and restarts the SPT computation from there, reusing the non-impacted region of the previous SPT. The further away the failure, the smaller the impacted subset and hence the bigger the iSPF computation gain compared to a full SPF. Last but not least, if the link does not belong to the previous SPT then the whole SPF computation may be skipped, as the old SPT is still valid.

We varied the size of the IS-IS network connected to our UUT from 500 to 10000 nodes and measured the duration of a full SPT computation for each network size. The obtained distribution showed a good linearity $(R^2 > 0.99)$ with the cloud size: Full-SPT(PRP2 processor) $\sim$45μs per node. A network of 700 nodes (large by current standards) is thus computed in the worst-case in $31.5ms$. In practice, the computation will often be much faster than this thanks to the iSPF optimization.

In conclusion, we have shown that the SPT computation is executed very fast (tens of milliseconds) and without any compromise on stability (dynamic throttle timers).

2.3.6 RIB and FIB update

The RIB/FIB update duration is linearly dependent with the number of modified prefixes[3].

[3]Routers have been improved to only modify the impacted prefixes. In the past, in some cases, the full FIB was rewritten [SG01]

Our UUT is once again used and a link failure is created such that all the 2500 prefixes from our topology are impacted by the failure. Packet generators create 11 streams, each of 1000 packets per second. The 11 streams are equally spread across the full table size (position1, 250, 500. . . 2500).

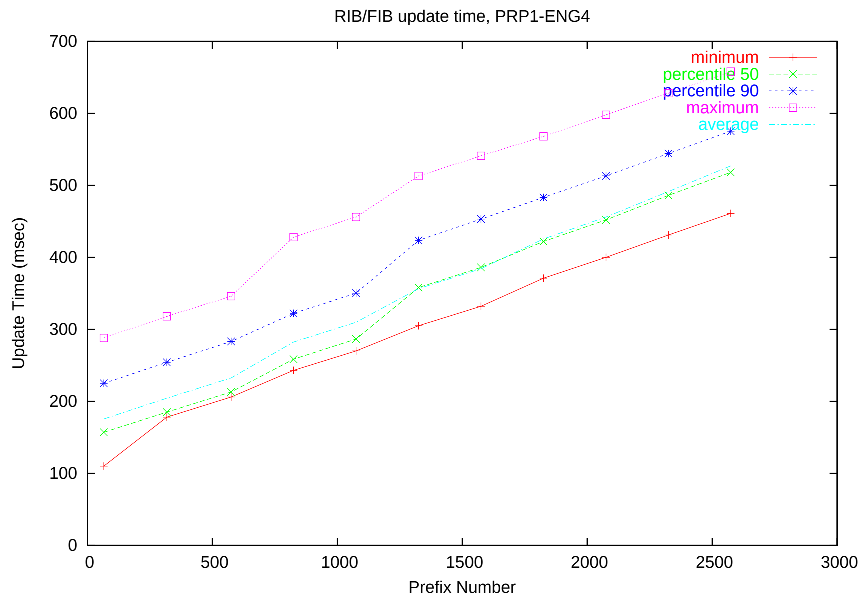


Figure 2.1: Minimum, Percentile-50, Percentile-90 and maximum RIB+FIB update time with PRP1 processor and Eng4+ linecards

We repeated the measurement 100 times and plot in figure 2.1 the percentile-0, 50, 90, 100 and the average update time. We repeated these tests with various processor speeds (GRP, PRP1, PRP2), various linecard types, local versus remote failure types and load balancing or not prior to the failure. Fig 2.1 provides the results when the UUT is equipped with a PRP1 processor, Eng4+ linecards, the failure is remote from the UUT and the UUT was not load-balancing before the failure.

As expected, the results primarily depend on the main processor performance (i.e. a PRP1 is twice more performant than the GRP. A PRP2 is faster than the PRP1) as this is the key bottleneck in the convergence process. The type of failure, the type of linecard, the load balancing state have a very moderate impact on the measured convergence and hence can be neglected in the remainder of this analysis. A linear regression on the percentile-90 indicated a cost per routing table update of $\sim 146\mu s$. This is the cost per RIB/FIB update.

Three approaches exist to minimize the RIB/FIB update component: network design rule to minimise the number of IGP prefixes, protocol and implementation optimisation to allow the prioritization of some prefixes above others during RIB/FIB update and finally the intrinsic optimization of the table management code. We will discuss here the two first approaches.

At the extreme, a designer could recognise that the only important prefixes that should be present in the IGP are those tracking premium content destinations (e.g., subnets with VoIP gateways) and BGP next-hops. All the other prefixes only track links interconnecting routers and this information could be advertised in iBGP. Unfortunately, many networks have not been designed like this as historically people did not care a lot about convergence. It is thus likely to see several thousands prefixes in the IGP of a large SP network while only a small fraction of them are really important. We thus face a problem where the RIB/FIB update component linearly scales by a number of several thousands while this number should in reality be much smaller.

Introducing prefix prioritization solves this problem: the important prefixes are updated first and hence the worst-case RIB/FIB update duration now scales based on a much smaller number (the number of important IGP prefixes as opposed to the total number of IGP prefixes). Prefix prioritization for IS-IS has been defined in [Cis03]. It introduces three priorities (high, medium, low) and guarantees that the routing table is always updated according to these priorities. A default heuristic classifies the /32 prefixes as 'medium' priority and the other prefixes as 'low' priority. The /32 prefixes are indeed likely more important than other prefixes as they characterize BGP speakers and tunnel termination services (i.e. L2VPN). Finally, a customization scheme based on IS-IS tagging is provided (e.g. subnet with VoIP gateways can be classified as 'high' importance and hence will always be updated first).

2.3.7 Distribution Delay

As we saw previously, the router implementation is optimized to allow for the parallel execution of the routing table update on the central CPU and the distribution of these modifications to the linecards.

The distribution of this information may be further optimized with for example the use of multicast transport between the central CPU and the linecard CPU's.

Reusing once again the same testbed, we measured the delta time between when a prefix is updated on the central CPU and when the related entry is updated on the LC. As expected, this 'distribution delay' was measured to be on average less than $50ms$ and in the worst-case less than $70ms$.

2.4 Simulation Model

The previous sections identified all the factors that influence the convergence time inside each router. In a large SP network, the total convergence time will also depend on factors that depend on the network itself. To evaluate those factors, we modified an OSPF implementation [Jac] for the SSFNet Simulator [Ren] to take into account the particularities of IS-IS and the white-box measurements presented earlier.

2.4.1 Router model

The measurements analysed in section 2.3 show that there are variations in the measured delays. Those variations are due to several factors such as the physical architecture of the router, the scheduler of the router's operating system, . . . To take those variations into account, we modified the simulator to use a delay within a $[min, max]$ range each time an event duration is considered in the simulator. The simulator randomly chooses a delay within the provided bounds. Although the measurements reveal a non-uniform distribution of the FIB update time, matching this distribution in the simulations would not provide more accurate simulation results because the position of a prefix in the RIB/FIB varies from one router to another, so that the time at which a prefix entry would be updated tends to be homogenized.

The first component of our model is the time required to detect the failure of a link. For a low delay link, we use the lab measurements presented in section 2.3. For long delay links such as trans-oceanic links, we randomly select one location and take into account the time to propagate the failure detection signal from this location to the two routers. In both cases, the two routers attached to a link will not detect its failure exactly at the same time.

Once a simulated router has detected a failure, it will originate a new LSP. We do not model the LSP generation timers in the simulator and allow the router to flood its LSP immediately. Doing this matches the recommended policy of not delaying the propagation of "bad news" about the state of links.

When a simulated router receives an LSP, it processes this LSP in [2,4]ms. Our router model supports both normal pacing and fast flooding as described in section 2.3.4.

After the arrival of a LSP indicating a failure, a simulated router needs to decide when to perform the SPT computation. We model the exponential backoff mechanism described in section 2.3.5. This mechanism is configured with three parameters : $spf\_initial\_wait$, $spf\_exponential\_increment$ and $spf\_maximum\_wait$.

In our simulations, we model the SPT computation time as a function of the number of nodes in the network with some jitter to take into account the other processes that may be running on the router's CPU. We only consider the full SPT computation and do not model the incremental variants. To model the time required to update the FIB of a router, we first compute the number of prefixes whose FIB entries have changed. The FIB update delay is then obtained by multiplying the number of FIB entries to be updated with the time required to update one entry. Our simulator models two types of FIB updates : *static* and *incremental*. With the static FIB update, the simulated router updates the FIB entry of each prefix after a recomputation of the SPT. This corresponds to routers such as those analysed in [SG01]. With the incremental FIB update, the simulated router only updates the

FIB entries for the prefixes whose nexthop has been modified after the recomputation of the SPT. This corresponds to the measurements discussed in section 2.3.6.

2.4.2 Convergence time

In section 2.3, we evaluated the convergence time of a router by sending packets through it and measuring the delay between the failure and the transmission of the first packet on a new interface after update of the FIB. This approach is not applicable for a large simulated networks because up to a few hundred of routers must be considered and sending packets is expensive in the simulator. Furthermore, sending packets as used by [PLM$^+$03] only samples the routers' FIBs at regular intervals.

To evaluate the convergence time of a network after a failure we use an approach similar to the one used by Kerapula et al. in [KCIB04]. When there are no failures inside the network, the routing is consistent, i.e. any router is able to reach any other router inside the network. After a link failure, the routers that were using the failed link need to update their FIB. Each router will update its FIB at its own pace, depending on the arrival time of the LSPs and its configuration. While the FIBs are being updated, the routing may not be consistent anymore. To determine the convergence time, we check the consistency of the FIBs of all simulated routers after the update of the FIB of *any* router. To do this, our simulator follows all the equal cost paths that a packet sent by a router S with D as destination could follow. If there is a forwarding loop for any $Source - Destination$ pair or if a router is forwarding packets on a failed link, then convergence is not reached. We define the *instant of convergence* as the last moment at which the routing becomes and remains consistent. Note that it is possible to find situations where the network converges transiently, then goes back into an inconsistent forwarding state, to finally reach a consistent forwarding state. This is the reason why we say we consider the last transition to a consistent forwarding state.

2.5 Simulation Results

In this section, we used the simulation model described in the previous section to first evaluate whether sub-second convergence after link and router failures is possible in large SP networks. We analyse the impact of the flooding component on the convergence time. We show that the RIB/FIB Update component is the determinant one and explain why fast-flooding is required to quickly converge after a router failure.

We use two representative, but very different SP topologies. The first one, GEANT, is the pan-European Research Network [GEA]. It connects all the National Research networks in Europe and has interconnections with research networks in other continents. GEANT is composed of 22 routers, 21 in Europe and one in New-York, USA. The network topology is highly meshed with a lot of redundancy in the core (Germany, Switzerland,France, UK, Netherlands) and fewer redundancy in the other parts of the network. Each POP is composed of a single router. It mainly contains continental links, which means that link delays are generally very low, except links that connect the network to the access router in New York.

The second studied network contains the backbone nodes of a worldwide Tier-1 ISP. The backbone of this network has about 200 routers routers in Europe, America and Asia. It is representative of a large commercial SP network. Each POP is usually composed of two core routers as well as several aggregation and access routers. In each POP, the core routers terminate the high bandwidth inter-POP links and are interconnected with redundant links.

To ease the comparison between the simulation results, we selected the same parameters for each network. Table 2.1 reports the values of all the relevant parameters. The only differences between the two networks are the SPF computation time that is function of the number of nodes and the number of prefixes advertised by each router, obtained from an IS-IS LSP trace.

Table 2.1: Simulation parameters

lsp_process_delay	[2,4]ms
pacing_timer	$\{6, 33, 100\}ms$
fast_flooding	on/off
spf_initial_wait	$\{10, 25, 50, 100\}ms$
spf_exponential_increment	$\{25, 50, 100\}ms$
spf_maximum_wait	$10000ms$
spf_computation_time	[20,30]ms in Tier-1 ISP
	[2,4]ms in GEANT
rib_fib_prefix_update_delay	[100,110] μs/prefix
rib_fib update type	incremental/static

2.5.1 IGP convergence after link failures

We begin our simulation study with link failures, the most frequent event that can occur in the topology of a network [MIB$^+$04]. For GEANT, we simulated the failures of all links. For the Tier-1 ISP, we simulated the failures of the 50 most loaded links.

When a link fails, the two routers attached to it detect the failure and originate a new LSP. Thanks to the two-way connectivity check [Ora90], a link is considered as having failed as soon as *one* of the two LSPs containing the link has been received by a rerouting router. This implies that the first LSP received after a failure is sufficient to allow any rerouting router to update its FIB.

We first used simulations to check that the sub-second IGP convergence target could be met in the GEANT network. We simulated the failure of each link. In figure 2.2, each curve shows the sorted simulated convergence times for the failure of each of the 36 links in GEANT. For the simulations, we set the $spf\_initial\_wait$ to $10ms$ or $100ms$ and evaluated the impact of the type of FIB update. The simulations show that the sub-second convergence after link failure is easily met in the GEANT network. This was expected given the propagation delays in the network, and its size. A closer look at the four curves in figure 2.2 shows that a lower $spf\_initial\_wait$ reduces the convergence time. The simulations also show the benefits of performing an incremental FIB update. The gain is important because when a link fails, only a small portion of the prefixes are reached via the failed link, and hence must be updated.

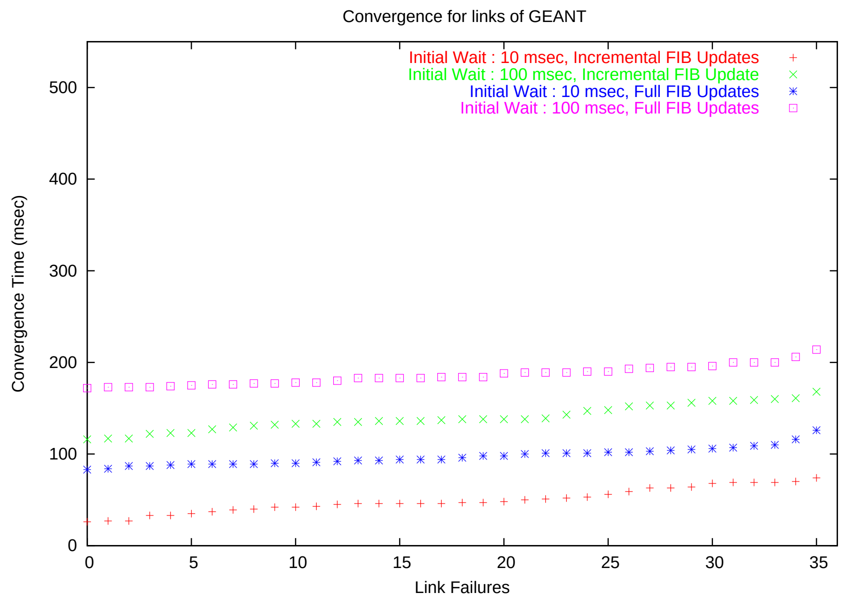


Figure 2.2: Convergence time for the link failures of GEANT, Initial Wait value set to $10ms$ and $100ms$, Static and Incremental FIB Updates

Achieving sub-second IGP convergence in Tier-1 SP networks is more challenging given the number of nodes, prefixes and the larger link delays found in a worldwide network.

Figure 2.3 shows that with all the considered parameters sub-second convergence is achieved. The simulations have been performed for the 50 links carrying the largest amount of source-destination paths. The overall convergence time in the Tier-1 SP is larger than in GEANT. This difference is mainly due to three factors. First, the link delays are larger in the Tier-1 SP. Second, the Tier-1 SP contains more IGP prefixes than GEANT. Third, the larger number of nodes in the Tier-1 SP leads to a longer SPF computation time. As for the simulations with GEANT, using a low $spf\_initial\_wait$ and incremental FIB updates reduces the convergence time. Note that in the Tier-1 SP, the benefit of using incremental FIB updates is much higher than in GEANT. This is because the total number of prefixes in the Tier-1 ISP is ten times larger than the number of prefixes in GEANT.

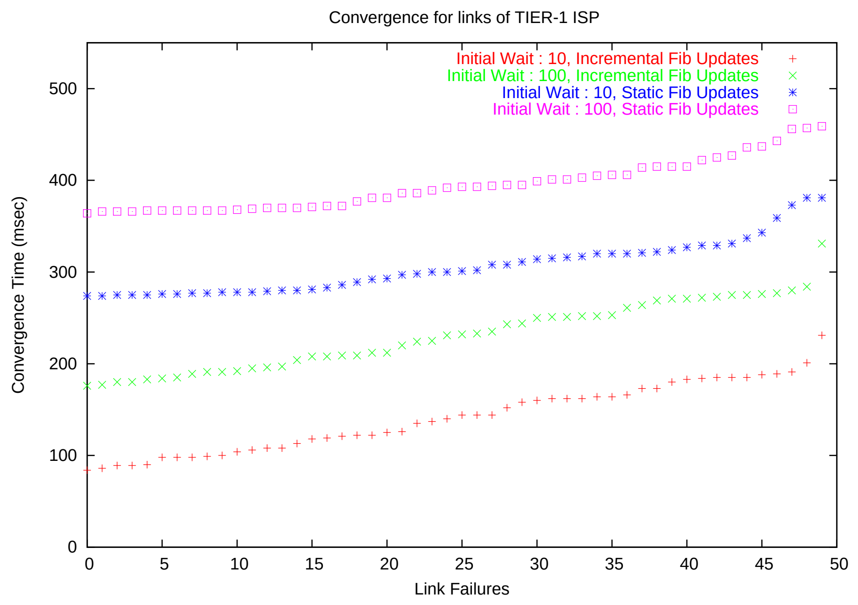


Figure 2.3: Convergence times for 50 link failures of Tier-1 ISP, Initial Wait value set to $10ms$ and $100ms$, Static and Incremental FIB Updates

To evaluate the impact of the topology on the IGP convergence, we performed simulations with several modifications to the topology of the Tier-1 ISP. We used the best simulation settings obtained from figure 2.3, i.e. $10ms$ $spf\_initial\_wait$ and incremental FIB updates.

First, to evaluate the impact of the link propagation delays on the convergence time, we built a new topology with all link delays set to one millisecond. Figure 2.4 shows that the IGP convergence times are only slightly reduced with this modification. This is mainly because first the SPF and FIB update times are the key factors in the IGP convergence of the studied network. Second, the IGP weights in this network, as in most SP networks, were set to favour rerouting close to the failure. This implies that rerouting occurs close to the failed link and hence the propagation time of the LSPs is a small component of the overall convergence.

Second, we modified the Tier-1 SP topology and set all link weights to one instead of the weight configured by the operator. The simulations show that this setting increases the IGP convergence time. This is because with such weights the rerouting routers can be farther from the failure than with the IGP weights configured by the network operators. Another consequence of this weight setting is that the FIB of more routers needs to be updated after each failure.

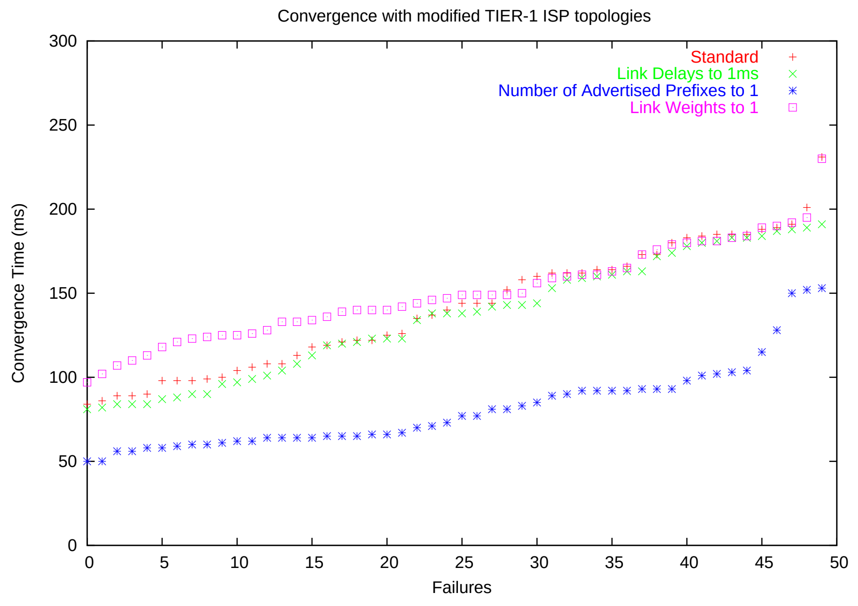


Figure 2.4: Convergence time for the link failures in the modified Tier-1 ISP, Initial Wait value set to $10ms$, Incremental FIB Updates

We obtained the most significant improvements in the convergence times by reducing the number of prefixes advertised by each router. When each router advertises a single prefix, convergence times are halved for nearly all the considered failures in the Tier-1 ISP. This shows that the number of advertised prefixes is one of the most important components of the convergence time.

Similar results were obtained with similar modifications to the GEANT topology.

2.5.2 IGP convergence after router failures

Besides independent link failures, ISP networks also need to face correlated link and router failures [MIB$^+$04]. To model such failures, we consider that all the links attached to a router fail at the same time. There are other types of SRLG failures (e.g. all links using the same optical fibre), but we did not have enough information on the physical structure of the simulated networks to correctly model those failures. For GEANT, we considered the failures of all routers while for the Tier-1 ISP we only simulated the failures of the 23 routers attached to the 50 links carrying the largest number of source-destination paths through the network.

The main difference between the failure of a single link and the failure of multiple links is that in the latter case, the first LSP received by a router is not always sufficient to describe the entire failure. In the case of a router failure, all the LSPs of the neighbours of the failed router might be necessary to correctly update the

FIB.

To evaluate the convergence time in the case of a router failure, we first consider a configuration that corresponds basically to IS-IS routers that have not been optimised for fast convergence : $33ms$ pacing timer without fast-flooding and static FIB updates.

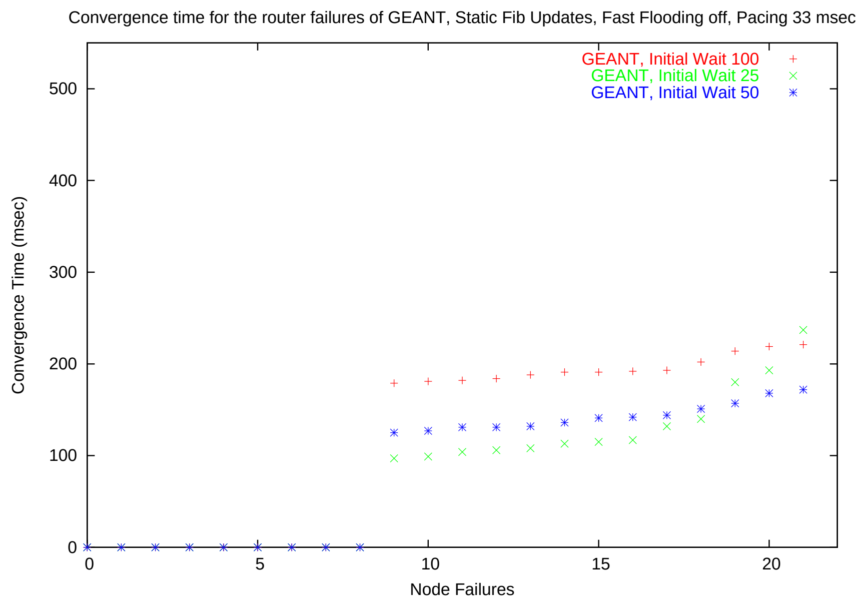


Figure 2.5: Convergence time for the router failures of GEANT, Static FIB Updates, Fast Flooding off, Pacing $33ms$

The simulations performed in GEANT (figure 2.5) show that this parameter setting allows to achieve sub-second convergence in case of router failures. In GEANT, the worse convergence time after a router failure was less than $250ms$. Surprisingly, the convergence time for some router failures was $0ms$. In fact, according to the IGP weights used by GEANT, those routers act as stub and do not provide any transit. When such a stub router fails, the reachability of the other routers is not affected. A closer look at the simulation results reported in figure 2.5 shows that the value of the $spf\_initial\_wait$ parameter does not have the same influence as with the link failures. For some router failures, the GEANT network can converge faster with a $100ms$ $spf\_initial\_wait$ than when this parameter is set to $25ms$. The simulation traces revealed that with a $25ms$ $spf\_initial\_wait$ some routers in the network had to update their FIB twice to allow the routing to converge. Those recomputations increase the convergence time.

We used the same parameter setting for the Tier-1 SP. Figure 2.6 shows that, in this case, the sub-second convergence is not achieved for router failures. We can see that for only 60% of the router failures, the convergence time is between 200 and $400ms$. For the other router failures, the convergence time can be as high as

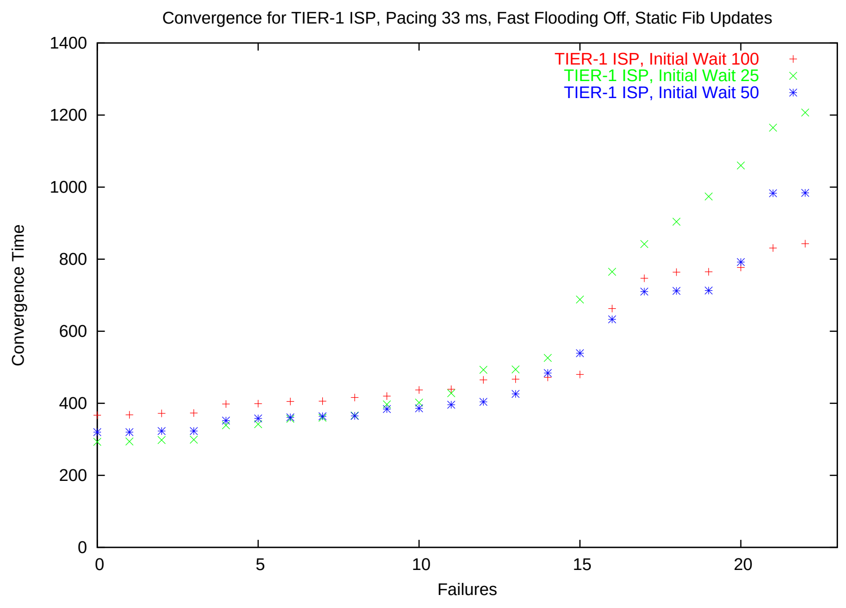


Figure 2.6: Convergence time for 23 router failures of Tier-1 ISP, Static FIB Updates, Fast Flooding off, Pacing $33ms$

$1400ms$. A closer look at the simulation traces revealed the reasons for those large convergence times.

The main problem is that some routers update their FIB before having received all the LSPs of all neighbours of the failed router. Unfortunately, this first update is not sufficient to allow the router to compute a correct FIB and a second, and sometimes third, update of the FIB is necessary. Given the number of prefixes advertised in the Tier-1 SP, those multiple static FIB updates explain around $660ms$ of the total convergence time. The remaining $600ms$ for some router failures are due to a cascading effect. With a single-threaded IS-IS implementation, a router cannot participate in the flooding of LSPs while it is recomputing its SPT or updating its FIB. With the standard pacing timer of $33ms$ and a $spf\_initial\_wait$ of $25ms$, a router can only receive one LSP from each of its direct neighbours before deciding to recompute its SPT. In some cases, corresponding to the left part of figure 2.6, those early LSPs are sufficient to correctly compute the final FIB and allow the network to converge. However, for the router failures corresponding to the right part of figure 2.6, the router spends almost $250ms$ to recompute its SPT and update its FIB. During this time, it does not flood LSPs and thus routers downstream do not receive updated LSPs and compute incorrect SPTs and FIBs. We verified this by analysing the traces and by setting the pacing timer to $100ms$. In this case, the convergence time was much larger. When the $spf\_initial\_wait$ is set to $50ms$ or $100ms$, the convergence time is reduced but still rather large.

To solve this problem, we must configure the routers to ensure that the routers only trigger their SPT computation once they have received all the LSPs describing the failure. This is possible by using the fast-flooding mechanism described in section 2.3.4.

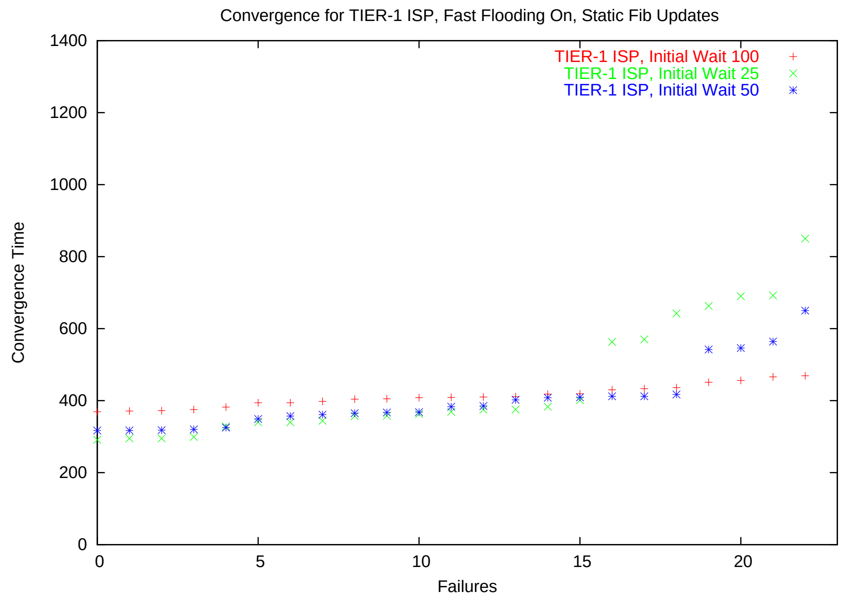


Figure 2.7: Convergence time for 23 router failures of the Tier-1 ISP, Static FIB Updates, Fast Flooding on

Figure 2.7 shows that when fast-flooding is used together with the static FIB updates, the sub-second convergence objective is easily met for all considered router failures in the Tier-1 SP. For 60% of the router failures (left part of figure 2.7), the $spf\_initial\_delay$ only has a limited influence on the convergence time. For the remaining router failures (right part of figure 2.7), a $spf\_initial\_wait$ of $100ms$ provides the lowest convergence time. With a $25ms$ or $50ms$ $spf\_initial\_delay$, the simulation traces reveal that some routers are forced to perform more than one update of their FIB, leading to a longer convergence time.

Besides the utilisation of fast-flooding, another possible modification to the configuration of the router would be to use incremental FIB updates. For the link failures, the improvement was significant.

Figure 2.8 summarises the simulations performed with fast-flooding and incremental FIB updates in the Tier-1 SP network. These simulations show that sub-second convergence is conservatively met also for the router failures in this network[4]. As explained earlier, the main benefit of using incremental FIB updates is to reduce the time required to update the FIB in all routers. When a failure affects only 10 prefixes on a given router, the FIB update time is around $1ms$ compared

[4]Note that the y scale changed in figure 2.8 compared to figure 2.7.

to the $220ms$ static FIB update time. This implies that even if a router triggers its SPT computation too early, it will block the LSP flooding for a shorter period of time. Furthermore, if a router needs to update its FIB twice, then fewer prefixes will be modified during the second update and this update will be faster.

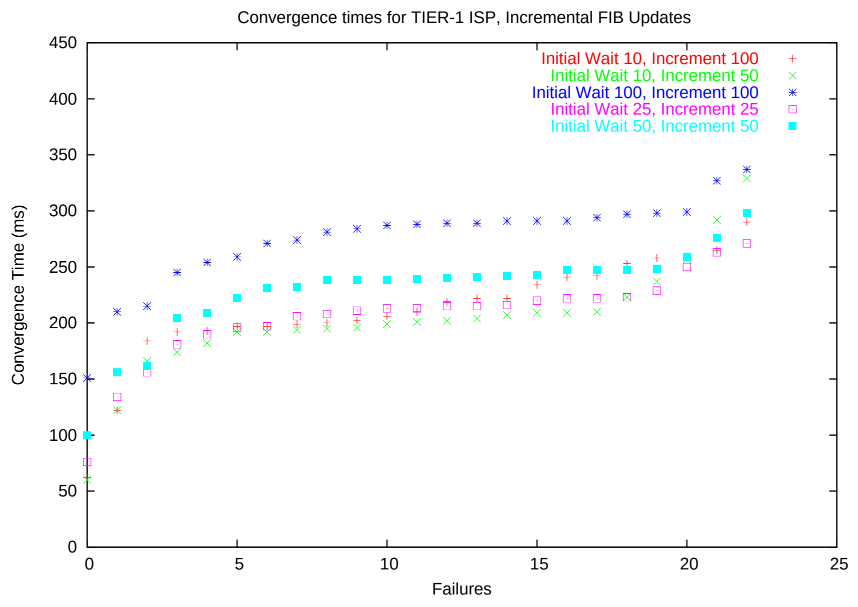


Figure 2.8: Convergence time for 23 router failures of Tier-1 ISP, Incremental FIB Updates, Fast Flooding on

We also used this simulation scenario to evaluate how the convergence time was affected by the configuration of the exponential backoff mechanism associated to the SPT trigger. The simulation results shown in figure 2.8 reveal that the most important parameter is the $spf\_initial\_wait$. As explained earlier, it should be set to ensure that for most failures, all LSPs have been received by all routers before the computation of the SPT. Our simulations do not indicate an optimal setting for the $spf\_exponential\_increment$. Finally, the setting of the $spf\_maximum\_wait$ depends on the acceptable CPU load on the routers during network instabilities.

We also performed simulations with fast-flooding and incremental FIB updates in the GEANT network. The simulation results reported in figure 2.9 show that a low $spf\_initial\_delay$ combined with a low $spf\_exponential\_increment$ provide the best IGP convergence times. A low $spf\_exponential\_increment$ is sufficient in this network given the small number of nodes and prefixes.

Our simulations clearly show that sub-second IGP convergence can be conservatively met in large SP networks with an appropriate tuning of the IGP configuration. First, the pacing timer should not be applied to urgent LSPs. Second, routers must flood urgent LSPs before recomputing their SPT and updating their FIB. Fast-flooding is thus recommended for fast convergence. Third, the router

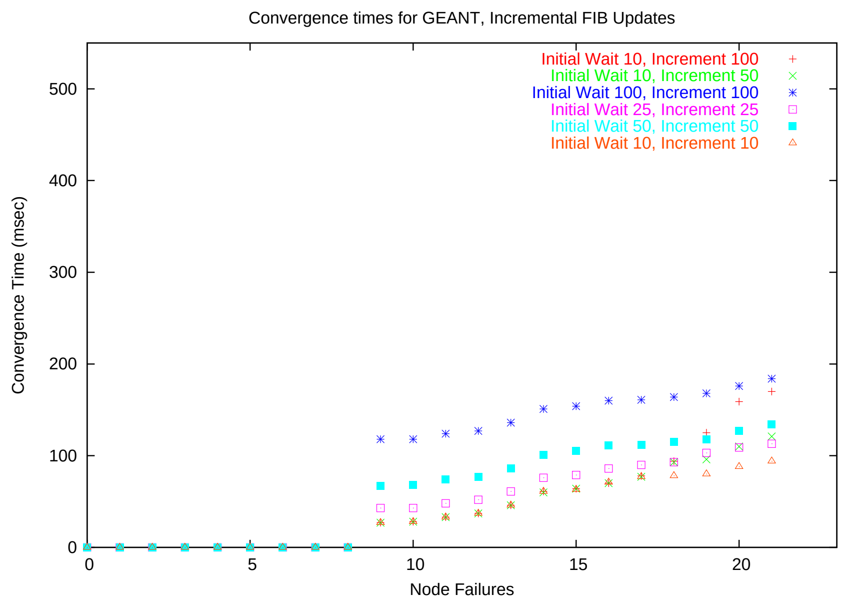


Figure 2.9: Convergence time for the router failures of GEANT, Incremental FIB Updates, Fast Flooding on

should need to modify the FIB entries only for the prefixes affected by the failure (incremental FIB Updates), and prefix prioritization should be used to let the most important ones be updated first. Fourth, using an incremental algorithm to update the SPT would also reduce the convergence time. Finally, in a large network, the configuration of the $spf\_initial\_delay$ on all routers in the network depends on the types of expected failures. If only individual link failures are expected, then the $spf\_initial\_delay$ can be set to a very low value such as $2ms$. If the network must converge quickly after router or SRLG failures, then our simulations show than in the Tier-1 SP network, a $spf\_initial\_delay$ of $50ms$ is appropriate. In operational networks, we would advice a more conservative value such as $150ms$. This value will allow the network to meet the sub-second IGP convergence objective with a sufficient margin to take into account various delays that could occur in the network and that cannot be accurately modelled in a simulator.

2.6 Guidelines to improve convergence time

Our simulations show that sub-second convergence is feasible in large SP networks. By taking care of some configuration guidelines, it is possible to bring an SP IP network back to a consistent IGP state within less than half a second, and within much less time in many cases. In this section, we summarize guidelines to achieve the fastest possible convergence with current IS-IS implementations. Also, we introduce potential ways to further improve the convergence time.

2.6.1 Failure detection

Failure detection should not be perfomed using Hello messages handled by the control plane. Indeed, tuning the sending rate of Hello messages to low values tends to increase CPU utilization on the control plane and hence harm the router perfomances. Instead, a failure detection mechanism built in the linecards themselves should be preferred. This can be done by using SONET alarms [Cisc], or by implementing BFD in the linecards [KW06]. When such mechanisms are used, control plane operations are only required when a failure is reported by the linecard.

2.6.2 LSP origination

Delaying the origination of LSPs describing an urgent topological change is harmful to the convergence time. However, allowing routers to generate and flood new LSPs at any rate can be harmful to the stability of the network, and trigger bursts of SPT recomputation and FIB udpates. Ideally, a router should prevent itself from spreading the instability of one of its adjacent links through the network. To do so, a link failure should directly trigger the flooding of an updated LSP, but a link-up event could be considered as a non urgent event triggering a delayed generation of an LSP.

Delaying the propagation of information concerning link-down events is obviously harmful for the forwarding of traffic, so that the stability goal can only be achieved by delaying the propagation of link-up events. However, delaying the utilization of a link being brought up could also be considered dangerous in the case of a congested network.

Requirements for the generation of LSPs could thus be summarized by the following .

- Avoid the delaying of bad news
- Avoid the propagation of flapping link-state information
- Avoid to keep a link artificially down for too long

A possible mean to achieve these goals is to associate a timer with each IGP link of a router. The value of this timer defines the time between the detection of a link-up event and the flooding of the LSP describing the link as up. The value of the timer associated with the link would then adapt to its stability. To do that, an exponential back-off mechanism similar to the one used to control the SPF recomputation rate could be used. When a link goes down, an updated LSP is directly flooded through the network, but the value of the timer associated with this link is increased, up to a configured maximum value. When the link is up and remains up for a configured value, the value of the timer is decreased. If the link returns into down state before the timer has elapsed, the timer is reset to its current back-off value.

2.6.3 LSP flooding

To achieve better convergence, pacing of urgent LSPs should be avoided. Implementations should distinguish urgent LSPs, describing a topological change and refresh LSPs, so that the relevant ones can be flooded without delay.

In order to avoid that bursts of LSPs originated by a set of routers or a misbehaving router continuously trigger SPT re-computations in all the routers, these could monitor the rate at which each router generates its own LSPs, and especially those describing non urgent good news. Alarms should be triggered once an irregular behaviour is detected. Note that this task could also be performed by an IS-IS/OSPF listener placed in each area of the network. Such monitoring can allow to use more aggressive timings while ensuring the stability of the network.

During the detailed analysis of our simulation results we found out that sometimes, routers delay the flooding of urgent LSPs because their IS-IS thread is currently recomputing an SPT, or performing a RIB/FIB update, so that it does not process and flood incoming urgent LSPs. To solve this problem, multi-threaded IS-IS processes could be used so that incoming LSPs could be processed and flooded during SPT recomputations and routing tables updates.

If we look closer at some long convergence time cases, we can see that they can sometimes be explained by topological constraints; the rerouting router is sometimes far from the failure. We indeed showed that the topology of a network, the delay of its links and their associated metrics can have an important impact on the convergence time of the IGP, by influencing the LSP propagation time component. One way to improve the convergence time in a network is thus to take it into consideration during the network design and evolution.

2.6.4 SPT computation

We consider the SPT computation component from two different aspect. The first aspect is the delay between the reception of a link-state change and the time at which the SPT recomputation is actually started. The second aspect is the computation of the SPT itself, i.e., the time between the beginning of the SPT recomputation and the time at which FIB updates are being sent to the linecards.

We showed that SRLG failures could lead to bad convergence times if too reactive configurations are deployed on routers that perform static FIB updates, within a single threaded IS-IS process. For those cases, conservative configurations help to obtain sub-second convergence times. However, these configurations tend to be suboptimal in the case of link failures, which are the most common events occuring in a network. Exponential back off mechanisms were introduced to face this problem. However, tuning the parameters of the back-off is not an easy task as the delay between LSPs arrivals at a router can vary according to the SRLG failure.

Ideally, a router should react immediately to a link failure when the network is stable, i.e. when no topological change occured for a long period of time. Then, if the router receives an LSP describing another link-state change, it should only update its FIB once it knows that it received all the LSPs describing the SRLG failure. Indeed, we noticed in our simulations that large convergence times can occur because routers have to perform 3 or more SPT re-computations and FIB updates to reach their post-convergence state. This is due to the fact that these were not aware of the whole topological change in the early reroute processes. In such scenarii, the exponential back-off can reach a state such that the delaying becomes very important and not required to perform the correct FIB update. To do that, the delay between the first and second execution of the rerouting process should be set according to a detailed analysis of the flooding behaviour in the considered network.

Once the re-computation of the SPT has been started, the results of the SPT computation should lead to FIB updates as soon as possible. Introduction of incremental shortest path computation helped in achieving this goal, although such optimizations sometimes take longer to complete, especially when the topological change is close to the root of the updated SPT. Another way to reduce the delay is to start feeding the FIB with updates **while** the SPT is being performed. Indeed, SPF finds shortest paths to further and further nodes, so that when a shortest path has been found with a distance x, all the shortest paths towards nodes at a distance $y < x$ have been found, and the corresponding FIB updates can be sent to the linecards.

2.6.5 RIB/FIB update

We also discovered by simulation that, in the core of an ISP backbone, where the network topology is highly meshed, rerouting routers are rarely far from the failure, and the number of prefix to update becomes the critical component.

Convergence time is thus greatly improved by reducing the number of prefixes that are advertised by the routers. The use of BGP next-hop-self option can help to achieve this goal, as it allows BGP speakers not to advertise prefixes associated with their peering links. However, not using next-hop-self favours fast inter-domain convergence upon BGP peering link failures. Though, BGP aggregate withdraws could be used in order to achieve fast inter domain convergence in BGP systems with next-hop-self enabled.

Similarly, the use of unnumbered links is also to be recommended in order not to advertise IP adresses associated with intra-domain links whose IP addresses are never used to send traffic.

2.7 Related work

The convergence of IGP protocols has been studied by various authors. Alaettinoglu et al. present in [AJY00] an analysis of the convergence of ISIS and explore changes to the ISIS specification and propose some improvements to routers implementations. Since the publication of this internet draft, the IETF and router manufacturers have significantly improved the convergence of IGP protocols. The fast ISIS hello timers proposed in [AJY00] have been replaced by a new protocol defined by the IETF : BFD [KW06]. In [AJY00], the SPF computation was considered as a major performance bottleneck. With the implementation of incremental SPF algorithms, this is not an issue anymore. Our measurements indicate that the main component of the IGP convergence, at the router level, is the FIB update time.

Shaikh and Greenberg present in [SG01] a detailed black-box measurement study of the behaviour of OSPF in commercial routers. Compared to this study, our measurements show in details the various factors that affect the performance of ISIS and take into account the multiple improvements to the ISIS implementations since the publication of [SG01].

Finally, Iannacone et al. evaluate in [ICBD04] the feasibility of providing faster restoration in large ISP networks. This feasibility was evaluated by using rough estimates of the possible IGP convergence time in a large ISP network. In this analysis, we have shown quantitatively that fast IGP convergence is possible by using measurement based simulations.

Cain proposed in [Cai00] to use the multicast forwarding facilities on routers to reduce the time required to flood link state packets. Our simulations show that ignoring the pacing timer for urgent link state packets is sufficient. Furthermore, in the Tier-1 ISP, the convergence time did not change significantly when we performed simulations with the link delays set to one millisecond.

2.8 Conclusion

In this chapter, we have presented a detailed study of all the factors that affect the convergence time of link state IGP protocols in large ISP networks.

We have first presented a detailed measurement study of all the factors that, on a single router, influence the convergence time. This time can be characterised as $D + O + F + SPT + RIB + DD$ where the detection time (D), the LSP origination time (O) and the distribution delay (DD) are small compared to our sub-second objective. The flooding time (F) depends on the network topology and thus on the link propagation delays. The SPT computation time depends on the number of nodes in the network, but can be significantly reduced by using an incremental SPT computation. Finally, the RIB time that corresponds to the update of the RIB and the FIB is the most significant factor in our testbed, as it depends linearly on the number of prefixes affected by the change. Note that by using prioritization techniques, it is possible to provide faster convergence for the most important prefixes.

We have then used simulations to evaluate the IGP convergence time in large ISP networks. Our simulations show that, in the case of link failures, a convergence time of a few hundred of milliseconds can be achieved by using a low initial wait timer for the SPF computation and incremental FIB updates. We also show that advertising fewer prefixes in the IGP significantly reduces the convergence time. When considering router or SRLG failures, the convergence time is only slightly larger provided that the pacing timer is disabled for urgent LSPs and that the initial wait timer is not too low. Handling SRLGs introduces a tradeoff.

Overall, our analysis shows that with current router technology sub-second IGP convergence can be conservatively provided without any compromise on stability. As explained in section 2.6, more precise, topology dependent, timing configurations and advanced implementation optimizations could help in reaching even lower convergence times.

In order to reach a much lower convergence time target, complimentary techniques whose convergence time does not depend on the topology should be used. Such techniques are discussed in the following chapter.

Chapter 3

IGP IP Fast ReRoute

Commercial IP networks supporting mission critical services must be capable of rerouting traffic very quickly in case of link failures. Due to its nature, the link-state IGP has limitations on its convergence time. We saw in chapter 2 that the IGP convergence time within an ISP can be up to a few hundreds of milliseconds, scaling with the the number of prefixes advertised in the IGP. In order to provide a faster restoration, the IGP must be complemented with other techniques. Such techniques prepare routers to the failure of the links they are directly connected to. They only require action from these directly connected routers to restore end-to-end connectivity through the domain. These are called Fast Reroute Techniques [VPD04].

In this chapter, we first describe in details the main IP-based Fast Reroute techniques that have been considered by the IETF : loop-free alternates [AZ07], U-turns [Atl06], protection tunnels [BFPS05] and “NotVia” addresses [BSP06]. We then use simulations to evaluate the network coverage of each technique by considering the network topologies of five very different Internet Service Providers (ISP) networks. Our simulations show that several techniques must be combined to allow an ISP to fully protect all its links. We also show that, when faced with distant link failures, the IP-based fast reroute techniques are as stable as the traditional MPLS-based techniques. Then, we discuss the practicality of these solutions, and their ability to protect LDP traffic. From this discussion, we conclude that loop-free alternates combined with NotVia addresses is the most realistic IP Fast Reroute technique. Finally, we propose improvements to the NotVia scheme that dramatically reduce the memory usage of such a solution.

3.1 Introduction

Some researchers have argued for achieving millisecond convergence after a failure [AJY00]. In practice, achieving this goal is difficult, due to the dynamics of the IGP, and the time required to update a FIB. As illustrated with white box measurements and simulations in chapter 2 and [FFEB05], and using black box measurements in [ICBD04], a more realistic estimate of the convergence time of a typical intradomain routing protocol in a large network is a few hundred of milliseconds.

For some mission critical services like voice or video over IP or PWE [XMP04], achieving a restoration time in the order of a few tens of milliseconds after a failure is important. In this chapter, we first present several techniques that can be used to achieve such a short restoration time. While most of the work on fast restoration has focussed on MPLS-based solutions [VPD04], recent work indicate that fast restoration techniques can also be developed for pure IP networks. Several researchers have proposed fast-reroute techniques suitable for IP networks [NST99, NLYZ03, LYN$^+$04]. Recently, the RTGWG working group of the IETF started to work actively on this problem [SB07] and several fast reroute techniques are being discussed.

This chapter is aimed at comparing these solutions and study their behaviour and their practical applicability in ISP networks. The chapter is organized as follows. First, we provide a detailed overview of fast restoration techniques suitable for pure IP networks in section 3.2. We focus on link failures that are the most common unplanned events in IP networks [MIB$^+$04, WJL03, SIG$^+$02]. We use MPLS-based techniques as a reference and focus our evaluation on pure IP-based techniques. In section 3.3 we evaluate by simulations how many links can be protected by each technique in large ISP networks based on their actual topology. This coverage is an important issue as some techniques cannot protect all links from failures. Then, we extend our simulations in section 3.4 to evaluate the stability of each protection technique. For this, we simulate all possible single link failures to determine whether each protection can remain active when distant links fail. In section 3.5, we discuss the applicability of the various solutions. We review the main issues of the proposals and we argue in favor of a solution combining Loop-free alternates and NotVia addresses. Next, we identify the key issues and potential solutions to achieve protection of LDP traffic using IP-FRR techniques. Finally, we discuss related work in section 3.6.

3.2 Fast reroute techniques

In this section, we describe the various protection techniques that can be used in IP networks. We first briefly present the protection techniques based on MPLS RSVP-TE as a reference since they are already deployed in several IP networks and then we present in details the IP-specific techniques under investigation at the IETF, that do not require the utilization of MPLS. Note that other solutions have been proposed in the literature [LYN$^+$04]. We do not discuss these in details in this chapter. We will comment them in section 3.6.

Before describing the protection techniques, it is necessary to introduce some terminology common to many protection techniques. We only consider protection techniques that are able to protect intradomain links, as not all of them have a node protection feature. The network that we consider is modelled as a graph $G(V, E)$

where V is the set of routers and E the set of links. We assume that the network is at least bi-connected. Providing an intra domain protection for a link whose failure disconnects the network is meaningless. Each link is modelled in the graph as two directed weighted edges. We will consider paths from a source router S towards a destination router D ($S \rightarrow \ldots \rightarrow D$). $Cost(P)$ is a function that returns the cost of path P, i.e. the sum of the metrics of the links that compose the path. $SP(S, D)$ is the (set of) shortest path(s) between source S and destination D. $SPT(S)$ is the shortest path tree rooted at router S. This is the tree capturing all the shortest paths from router S to all the other routers of the network. Similarly, $rSPT(S)$ is the reverse shortest path tree rooted at router S, i.e. the tree capturing all the shortest paths from the routers of the network to router S. $N(I)$ is the set of direct neighbours of router I.

To protect a link, we will need to consider the two directed edges that compose the link. A router can only protect or quickly reroute the packets that it sends on a link, not the packets that it receives on this link. When considering the protection of link $I \rightarrow J$ between routers I and J, we will call router I the protection router and J the primary nexthop.

An IP-based fast-reroute technique aims at recovering the reachability of destinations by the sole reaction of the protection router, i.e., the head-end of the failing link. When this router detects the failure of one of its protected links, it must quickly update its Forwarding Information Base (FIB) to protect the affected packets by sending them over another link. Note that in current router architectures, the FIB is replicated in the linecards, so that the control plane of the router has to download FIB updates to each linecard. To perform this update within the sub-50 milliseconds target, some tuning of the FIB is required.

Conceptually, the FIB of a router can be considered as being equivalent to a two-column table (figure 3.1). On each line of the table, the first column indicates the destination prefix and the second the outgoing interface (OIF) to be used to forward a packet towards a destination. The OIF can be either a physical interface or a logical interface such as a tunnel.

This kind of FIB organization is not suitable to allow a router to quickly recover the reachability through the network after a failure. Its main drawback is that each FIB entry affected by the failure must be updated after the failure. As shown in chapter 2, high-end commercial routers require around 110μsec to update a FIB entry [FFEB05, Fil05]. Given the large number FIB entries on the routers in large ISP networks, it would be impossible to update all FIB entries affected by the failure one after the other within the 50ms target.

A second possible organization of the FIB is to rely on pointers. Such an organization is illustrated in the middle part of figure 3.1. The FIB is now conceptually composed of two tables. The first contains all destination prefixes and pointers to the outgoing interfaces used to reach those destinations. The second table contains one flag and two data structures. Each datastructure contains all the information required by the router to forward packets over this interface (e.g. layer 2-framing to be used, layer-2 address of nexthop, encapsulation in case of virtual interfaces,

. . .). When the flag is set to “Up”, the first outgoing interface is used and otherwise the second.

The main advantage of this FIB organization compared to the previous one is that when a link fails, it is possible to quickly reroute all the destinations affected by the failure by simply changing the flag of the corresponding line in the interface table. Also, the memory requirements of this technique are low as they scale with the number of interfaces on the router.

A third possible organization of the FIB, illustrated in the bottom part of figure 3.1 is to store pointers to the primary as well as to the secondary outgoing interfaces used to reach each destination in a first table. In a second table, we only store the datastructures required to forward the packets over this interface and a flag corresponding to the current interface state. When a packet is to be forwarded using that FIB design, the router performs a lookup on the packet destination, it then selects a primary outgoing interface for this packet. When the flag of this interface is found to be down in the second table, the router uses the backup interface that is stored in the first table.

The main advantage of this FIB organization compared to the previous one is that two destinations reached via the same outgoing link can be protected by using distinct protections. We will see in section 3.5.2 that it also eases the use of IP Fast Reroute to protect LDP traffic. Note that with this third solution, the FIB memory requirements to support Fast Reroute scale with the number of prefixes stored in the FIB.

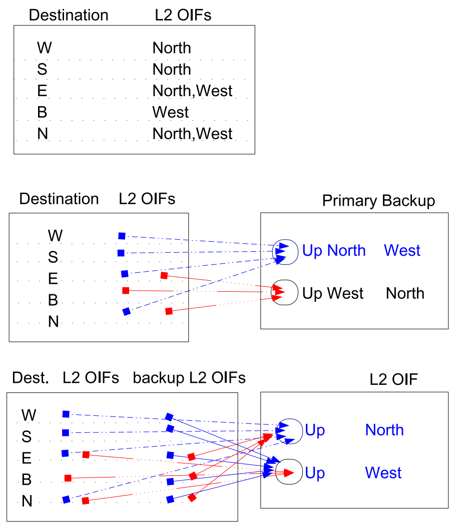


Figure 3.1: Three different organizations of the FIB for router A

To illustrate the various fast restoration techniques, we will use the six-routers topology shown in figure 3.2.

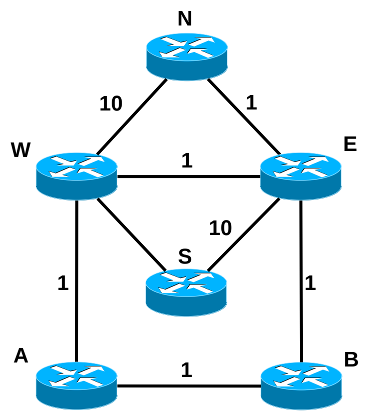


Figure 3.2: Simple network topology

3.2.1 MPLS-based protection techniques

The first technique proposed, implemented and deployed to quickly reroute IP packets when a link fails is to use MPLS [VPD04]. Several mechanisms have been proposed in the last years. In MPLS networks, two techniques are possible to protect Labelled Switched Paths (LSPs) : fast reroute and bypass tunnels. In both cases, each segment of a LSP is protected by pre-establishing a secondary LSP that is disjoint form the resource being protected and merges with the primary LSP downstream of the protected resource.

In IP networks that are not using MPLS to forward IP packets, it is possible to use MPLS only to provide protection [SP03]. In this case, pure IP forwarding is used when the network is stable and MPLS is only used to transiently forward the packets around the failed links. Formally, a MPLS protection tunnel that protects link $I \rightarrow J$ is defined as the shortest path between routers I and J on the network topology after having removed link $I \rightarrow J$. A router can easily compute the path of the protection tunnel required to protect each link by using the Dijkstra algorithm on the reduced topology. The MPLS protection LSP is established by the protection router by using RSVP-TE.

In figure 3.2, router W could protect link $W \rightarrow E$ by establishing an MPLS LSP over the $W \rightarrow N \rightarrow E$ or the $W \rightarrow S \rightarrow E$.

If the network is bi-connected, then those MPLS LSP can be used to protect any single link failure. Thus the coverage of this technique is 100%. Its main drawback is that it forces the ISP to deploy both MPLS and RSVP-TE only to support fast restoration.

3.2.2 Equal Cost Multipath (ECM)

A first IP-based solution to protect a link that could fail is to ensure that a router is not using *only* this link to reach a given destination. One possibility is to install parallel (and preferably physically disjoint) links between each pair of routers. This technique is used by some ISPs for some important links, but using it to protect all links would be too expensive.

With current implementations of link-state routing protocols (OSPF and ISIS), another solution is possible [AZ07]. When a router uses the Dijkstra algorithm to compute its shortest-path tree, it may find several equal cost paths to reach destination d. As those paths have exactly the same cost, the router may select any of them to reach destination d. Today's routers are also able to install all those paths in their forwarding table and rely on a hashing algorithm to select the packets that must be sent over each path. From a restoration viewpoint, the main advantage of using ECM is that when one path becomes unavailable, then if the other paths are completely disjoint, they are still active. Thus, the traffic can be quickly protected by updating the forwarding table and simply removing the entries corresponding to the failed link. Discussions with large network operators indicate that ECM is often enabled and used in their network to better load-balance the traffic [ICBD04].

Formally, the directed link between routers I and J, $I \rightarrow J$, is protectable by link $I \rightarrow K$ for destination d if $Cost(I \rightarrow J \ldots d) = Cost(I \rightarrow K \ldots d)$.

In figure 3.2, router A can use two paths to reach destinations E and $N : A \rightarrow W \rightarrow E$ and $A \rightarrow B \rightarrow E$. If router A detects a failure of link $A \rightarrow B$, it simply stops using the second path and the packets towards E and N are no longer affected by the failure.

However, the main drawback of relying on ECM is that it cannot protect all destinations and all links. This is the case in our example for destination B. When link $A \rightarrow B$ fails, router A must wait until the convergence of the IGP to be able to use the alternate path via routers W and E. More generally, relying on ECM to achieve fast restoration in case of failure is very impractical. An ISP that relies on such a technique has to design its topology to have at least two Equal Cost Shortest Paths from each node to all the other nodes. ECMP is used by some ISPs for some important links, but protecting all the links of a topology using ECMP would be too expensive.

3.2.3 Loop-free alternates (LFA)

Principle

Besides parallel links and ECMP, the first IP-based protection technique being considered within the IETF is the utilization of loop-free alternates [AZ07]. If router I is using link $I \rightarrow J$ to reach destination d, then a loop-free alternate is a direct neighbor, say router N, of router I that is able to reach destination d without using link $I \rightarrow J$. This means that if link $I \rightarrow J$ fails, router I can deviate the packets towards d to N instead of J. These deviated packets will reach d and will not loop on link $I \leftrightarrow N$. Formally, a loop-free alternate for destination d at router N is defined in [AZ07] as a router N such that $Cost(N \rightarrow \ldots d) < Cost(N \rightarrow \ldots I) + Cost(I \rightarrow \ldots d)$. Since routers use

Link	Loop-free alternates
$W \rightarrow E$	N
$E \rightarrow W$	S
$S \rightarrow W$	E
$N \rightarrow E$	W

Table 3.1: Links protectable via a loop-free alternate in figure 3.2

shortest path routing, an equivalent condition is that $(I \rightarrow J) \notin SP(N, d)$.

To understand the utilization of loop-free alternates, let us consider link $A \rightarrow W$ in figure 3.2. This link is used by router A to reach destinations W, S, E and N. When this link fails, router A can quickly reroute the affected packets by updating its FIB and sending them to router B. To understand this protection, we have to consider the SPT computed by router B. This router uses link $B \rightarrow E$ to reach destinations E and N. For destination E, router B is a loop-free alternate since $Cost(B \rightarrow E) < Cost(B \rightarrow A) + Cost(A \rightarrow W \rightarrow E)$ $(1 < 1 + 2)$. Unfortunately, for destinations W and S, due to the utilization of ECM, router B is not a loop-free alternate.

If we consider link $W \rightarrow E$ in figure 3.2, we can find several loop-free alternates. First, routers N and S are loop-free alternates to reach respectively destinations N and S. Second, router A is a loop-free alternate to reach destination B. Unfortunately, there is no loop-free alternate that can be used to reach router E after the failure of link $W \rightarrow E$.

Coming back to figure 3.2, it is easy to see that router E can act as a loop-free alternate for all destinations using link $S \rightarrow W$ as router E does not use this link to reach any destination. 4 directed links carrying traffic can be fully protected by using loop-free alternates. Table 3.1 provides for each fully protectable link the possible loop free alternates.

LFA modes

Loop Free Alternates can function on a per destination basis or on a per link basis. In **per destination** mode, a router I protecting the link $I \rightarrow J$ tries to find a neighbor N that does not use the link to reach **each** destination that I initially reaches via $I \rightarrow J$.

In **per link** mode, a router I protecting the link $I \rightarrow J$ tries to find a neighbor N that does not use the link to reach **all** the destinations that I initially reaches via $I \rightarrow J$.

The per destination mode offers a better coverage, as sometimes a per link loop free alternate does not exists for a given link although there are loop free alternates for some destinations reached via the protected link. For example, in figure 3.2, there is no per link LFA to protect link $A \rightarrow W$, because the link is used by B to reach W. However, B is a valid LFA for destination E. Note that if there does not exist a per link LFA for a link $I \rightarrow J$, then I will not find a per destination LFA for each destination reached via this link. Indeed, if there is no per link LFA, all the neighbors of I have link $I \rightarrow J$ in their SPT, so that there is no per destination LFA for at least destination J.

These two modes introduce a tradeoff between coverage and computational complexity. Indeed, under per prefix mode a node R tries to find a neighbor N that does not use a particular link $R \rightarrow X$ to reach each destination d by looking at the paths from N to d, in $SPT(N)$. Under per link mode, only the presence of the link $R \rightarrow X$ in $SPT(N)$ has to be checked.

3.2.4 U-turns

As illustrated above in figure 3.2, a loop-free alternate does not always exist in real networks. A closer look at ISP topologies showed then when there is no loop-free alternate neighbour to fully protect a link, there is often a router two hops away that does not utilize the link to be protected. This motivated the introduction of U-turns in [Atl06].

To understand intuitively the behaviour of a U-turn, consider again the network topology shown in figure 3.2. It was shown above that link $A \rightarrow B$ cannot be completely protected by a loop-free alternate. When link $A \rightarrow B$ fails, if router A decides to forward all its packets on the $A \rightarrow W$ link, router W will forward the packets towards E correctly, but will unfortunately forward the packets destined to B on the $W \rightarrow A$ link. This is unfortunate as if router W, being informed about the failure of link $A \rightarrow B$, had forwarded the packets to router E instead, then the link would have been protected.

More precisely, a neighbour U of router I can act as a U-turn to protect link $I \rightarrow J$ if one of its neighbours, say router R, does not utilise link $I \rightarrow J$ inside its SPT.

$U \in N(I)$ and $R \in N(U)$ and $(I \rightarrow J) \notin SPT(R)$

When router I does not find a loop-free alternate to protect link $I \rightarrow J$, it can compute the SPT of the neighbours of its neighbours to determine whether a U-turn is possible. This increases the number of SPT to be computed by each router after each topology change. To serve as a U-turn alternate, router U must be able to support two types of forwarding. When the network is stable, router U uses its normal FIB to forward packets. For the packets affected by the failure that are *u-turned* by router I, router U must detect that these are affected packets and forward them directly to the alternate router, router R without using its normal FIB. Several solutions are possible to detect that a packet was affected by a failure [Atl06]. The first solution is that router U performs an RPF-like check for each received packet. If the packet is currently following the normal SPT, it is forwarded by using the normal FIB. Otherwise, the packet is considered to be an affected packet and is forwarded directly to the alternate (router R). A second possible solution would be that router I explicitly marks the affected packets, for example by using a special DiffServ Code Point.

Link	U-turns
$W \rightarrow A$	$E \rightarrow B$
$W \rightarrow S$	$E \rightarrow S$
$A \rightarrow W$	$B \rightarrow E$
$A \rightarrow B$	$W \rightarrow E$
$E \rightarrow N$	$W \rightarrow N$
$E \rightarrow B$	$W \rightarrow A$
$B \rightarrow A$	$E \rightarrow W$
$B \rightarrow E$	$A \rightarrow W$

Table 3.2: Links protectable via a U-turn in figure 3.2

Compared with the loop-free alternates, the main drawback of the U-turns is that they require a cooperation of the neighbours and some modifications to the router's interfaces.

In the topology shown in figure 3.2, 8 directed links carrying traffic that cannot be protected by using loop-free alternates can be protected by using U-turns. Table 3.2 provides for each protectable link the possible U-turns.

3.2.5 Protection tunnels

The loop-free and U-turn alternates discussed in the previous section are not sufficient to provide a full coverage in large networks. This coverage can be improved by using IP tunnels as proposed in [BFPS05]. Besides MPLS that was discussed earlier, several tunnelling schemes are used in IP networks : L2TP [LTG04], GRE [FLH$^+$00], IP in IP, ... These tunnelling schemes can be used to create virtual links between routers. While in the past packet encapsulation and decapsulation was performed by the central CPU with a limited performance, interfaces on current high-end routers are now able to encapsulate and decapsulate tunnelled packets at wire speed.

IP tunnels can be used to efficiently complement the loop-free alternates described above. With a loop-free alternate, the packets affected by a failure where rerouted to a neighbour that does not utilise the failed link to reach the affected destinations. By using tunnels, it is possible to expand the loop-free alternate to utilise virtual neighbours. The principle of the utilization of protection tunnels can be sketched as follows. To protect directed link $I \rightarrow J$, router I must be able to send the affected packets to a router that is not currently using the failed link.

For this, router I needs to find a router N that is reachable without using the link to be protected and that is also able to forward packets to any destination without using link $I \rightarrow J$. Formally, router N is such that :

$(I \rightarrow J) \notin SPT(N)$ and $(I \rightarrow J) \notin P(I, N)$

When this condition holds, router I can encapsulate the affected packets and send them inside a tunnel towards router N. When N receives such an encapsulated IP packet, it decapsulates it and forwards it according to its current FIB. As

$SPT(N)$ does not contain $I \rightarrow J$, it does not reach the destination of the packet via $I \leftrightarrow J$. Furthermore, router N is neither using the opposite direction of the failed link. Indeed, a destination that was previously reached by I via $I \rightarrow J$ cannot be reached by J via $J \rightarrow I$ otherwise there is a persistent loop.

In figure 3.2, directed $A \rightarrow B$ can be protected by using a tunnel between router A and router S. This is a valid protection tunnel because : first router A reaches the tunnel endpoint, S without using the protected link. Second, $(A \rightarrow B) \notin SPT(S)$ and thus router S will not return the packets received via the tunnel over the protected link. In this topology, the forwarding of the packets is not optimal while the protection tunnel is active since the protected packets will be transmitted over both directions of link $W \leftrightarrow S$. As the protection tunnel will only be active during the IGP convergence, this is not a significant problem.

A method to compute the tunnel endpoint to protect a link was proposed in [BFPS05]. To protect link $I \rightarrow J$, router I must first determine the routers that it can reach without using this link. This can be easily obtained by computing $SPT(I)$ and pruning from this tree all the routers that are reached via link $I \rightarrow J$. This set is called the *F-space* in [BFPS05]. The set of possible tunnel endpoints is the set of routers that are able to reach router J without using link $I \rightarrow J$. This set can be computed as $rSPT(J)$ pruned from link $I \rightarrow J$ and all the routers that reach router J via this link. This is called the *G-space* in [BFPS05]. The set of candidate tunnel endpoints is then the intersection between the F-space and the G-space. If the set contains several routers, then a criteria must be defined to select the best one. If the set is empty, then no protection tunnel can be established to protect this link.

3.2.6 Protection tunnels with directed forwarding

In some topologies, it is not possible to find a tunnel endpoint to protect each link. A closer look at these cases reveals that often, although there is no intersection between the *F-space* and the *G-space*, a router, say F, of the *F-space* is a neighbour of a router, say G, in the *G-space*. Neither of those routers can be used as tunnel endpoints. Router F can receive packets from the protection router without using the protected link, but it uses the protected link. Router G on the other hand does not use the protected link but the protection router uses the protected link to reach it. Thus, router F can receive encapsulated packets but cannot forward them by using its FIB. Router G can use its FIB to forward the packets, but cannot receive the encapsulated packets from the protection router. A protection tunnel can be established by using both routers provided that once router F receives an encapsulated packet it forwards it directly to router G without using its FIB. This type of *directed* forwarding can be achieved by labelling the encapsulated packets that router F should forward to router G.

This labelling can be inserted in the encapsulated packets in various ways. The first solution is to use MPLS over GRE or MPLS over IP [RR04]. In this case, when router F receives an encapsulated packet, it first decapsulates it and them processes the MPLS label and forwards it to router G. A similar labelling is possible by relying on the *Key* field of the GRE encapsulation. In operational networks, the choice of the encapsulation scheme to use will depend on the router capabilities. To deploy such a scheme, minor extensions to IS-IS and OSPF will be required to allow each router to advertise the supported type of protection tunnel and to associate a label to each of its neighbours [BFPS05].

The main advantage of the protection tunnels with directed forwarding compared to the normal protection tunnels is that when all the link metrics are symmetrical, it can be proved (see theorem 3.2.1that each directed link can be protected by such a tunnel provided of course that the network is at least bi-connected.

Theorem 3.2.1 *In a network topology G(V,E), bi-connected, with symmetrical edge weights, a protection tunnel with directed forwarding can be used as an alternate path to protected any edge $S \rightarrow D$ of the topology.*

Let the set of nodes reached via $S \rightarrow D$ in $SPT(S)$ be $Nodes_{S,S \rightarrow D}$.

1. link $(X \rightarrow Y) \in P(A, B) \Rightarrow (Y \rightarrow X) \in P(B, A)$.

   This is a classical property of the shortest path tree in a graph with symmetric weights.

2. link $(X \rightarrow Y) \in SPT(N) \Rightarrow (Y \rightarrow X) \notin SPT(N)$

3. As $G(V, E)$ is bi-connected, the network remains connected after the failure of $S \rightarrow D$.

4. From 3, $\exists N \notin Nodes_{S,S \rightarrow D} : \exists (N \rightarrow N') : N' \in Nodes_{S,S \rightarrow D}$

   Indeed, to reach the nodes that were previously reached via $S \rightarrow D$, there must be a link connecting a node N that is not reached via $S \rightarrow D$ and at least one node N' that was reached via $S \rightarrow D$ in $SPT(S)$.

5. From 1, $N' \in Nodes_{S,S \rightarrow D} \Rightarrow D \rightarrow S \in SPT(N')$

6. From 2 and 5 : $S \rightarrow D \notin SPT(n')$

   The protection tunnel with directed forwarding is thus : $S \rightarrow \ldots N \rightarrow N'$ ∎

Unfortunately, real networks do not always use symmetrical IGP weights. This asymmetry may be intentional, e.g. due to the utilization of IGP weights optimized for traffic engineering [FRT02] or due to a configuration error. In such a network, it may be impossible to protect a link by using a tunnel with directed forwarding. This is illustrated in figure 3.3. In this network, all links have a weight set to 1 except directed link $S \rightarrow T$, which has a weight of 10 and $Z \leftrightarrow R$, which has a weight of 4. R can only reach Z without using $R \rightarrow T$, as it has an ECM path to Y via the protected link. Z uses link $R \rightarrow T$ and thus cannot be a protection tunnel endpoint. Y cannot be used as a directed forwarding tunnel ($R \rightarrow Z \rightarrow Y$) endpoint as Y also uses $R \rightarrow T$ to reach T.

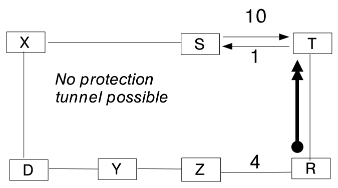


Figure 3.3: Example topology where protection tunnels with directed forwarding cannot protect all links

Router	Paths to E	Paths to E_W
N	$N \rightarrow E$	$N \rightarrow E_W$
S	$S \rightarrow W \rightarrow E$	$S \rightarrow W \rightarrow A \rightarrow B \rightarrow E_W$
W	$W \rightarrow E$	$W \rightarrow A \rightarrow B \rightarrow E_W$
A	$A \rightarrow W \rightarrow E$, $A \rightarrow B \rightarrow E$	$A \rightarrow B \rightarrow E_W$
B	$B \rightarrow E$	$B \rightarrow E_W$

Table 3.3: Paths chosen when *NotVia* addresses are used

3.2.7 NotVia addresses

A last protection technique was proposed recently in [BSP06]. This solution can be considered as an extension of the protection tunnels described earlier, but it requires a cooperation among all the routers of the network. Intuitively, the idea of this solution is that to protect link $I \rightarrow J$, router I should be able to send the affected traffic inside a tunnel towards a special address of router $J : J_I$. This address is a special *NotVia* address. Its semantics is that all routers of the network must have computed their FIB such that they *never* use link $I \rightarrow J$ to forward packets towards destination J_I.

In figure 3.2, link $W \rightarrow E$ could be protected by using *NotVia* addresses as follows. Table 3.3 shows how paths selected by all routers to reach router E via address E and via address E_W (i.e. without using link $W \rightarrow E$).

This solution requires a cooperation of all the routers inside the network. A router with n neighbours will advertise inside its link state packets one *NotVia* address for each of its neighbours. Upon reception of such a link state packet, each router will compute a special FIB entry for each *NotVia* address. This FIB entry is obtained by computing the router's SPT on the network topology without the link corresponding to the *NotVia* address.

In theory, this means that after each topology change, each router in the network should recompute one SPT for each *NotVia* address. If all links are protected by using such addresses, then each router would have to recompute as many SPTs as there are (directed) links in the network. Such a computation is not feasible in large networks, but [BSP06] reports that by using incremental-SPF algorithms, the actual cost of this computation in several ISP topologies was similar to five to thirteen times the cost of computing the SPF with the normal Dijkstra algorithm. Compared to the other techniques, the main advantage of the *NotVia* addresses is that this solution is applicable to all links, even in asymmetrical networks.

3.3 Network coverage of the IP-based fast reroute techniques

As described briefly in the previous section, a potential issue with the IP-based fast reroute techniques is that several may be required to fully protect all links in networks. All protection techniques are not equivalent. From an implementation viewpoint, the *loop-free alternates* is the simplest solution. The *U-turns* allow to protect more links than the loop-free alternates, but they require changes to the routers' FIBs. The *protection tunnels* require some computation to select the tunnel endpoint at the protecting router and use encapsulation. Finally, the *NotVia* addresses force each router to compute one SPF per *NotVia* address. Our objective in this section is to determine the *network coverage* of IP-based fast reroute techniques, i.e. the number of links that can be actually protected for a given network topology.

To evaluate the suitability of the IP-based fast reroute techniques, we implemented a simulation tool that is able to analyse any network topology. Our simulator models how a router would select an IP-based fast reroute technique to protect its links. For each link in the network topology, the simulator performs the following tests. First, its tests whether the link can be protected by one or more loop-free alternates. If this technique cannot be used, the simulator tests whether U-turns are available. It there is no U-turn, the simulator tries to find an endpoint for a protection tunnel. Usually, more than one router can be used as a tunnel endpoint. The simulator selects the best endpoint as the endpoint with the shortest IGP distance from the protecting router. If no protection tunnel is possible, the simulator tries to find a tunnel endpoint for a protection tunnel via directed forwarding. Finally, the simulator computes the path that would be selected by using *NotVia* address. For comparison purposes, the simulator also computes the bypass tunnel that would be established by using RSVP-TE with MPLS-based protection techniques. We applied the basic cSPF variant to compute the MPLS-FRR protection. That is, we consider that the MPLS tunnel is established along the shortest path from the head-end of the link towards the tail-end of the link, with the constraint that the protected link cannot be used.

3.3.1 Small networks

We first used the simulator to evaluate the network coverage of the IP-based fast reroute techniques in regular network topologies. We considered 4 topologies.

The first topology that we consider is the *ring*. Each router is connected to two neighbours and all links have the same cost. This topology is the worse topology from a fast-reroute viewpoint. There is no loop-free alternate, no U-turn and no protection tunnel. However, all links can be protected by using protection tunnels with directed forwarding. This tunnel with directed forwarding is illustrated in the upper-left part of figure 3.4.

The second topology that we consider is the *double-star* (upper right part of figure 3.4. It is typical of small ISP networks that are composed of two redundant core routers and remote POPs containing two routers. The two routers of each POP are directly connected and also attached to the tow core routers. In this topology, when all IGP weights are the same, all links are protectable by using loop free-alternates. For a link from a POP router to a core router, the loop-free alternate is the other router in the same POP.

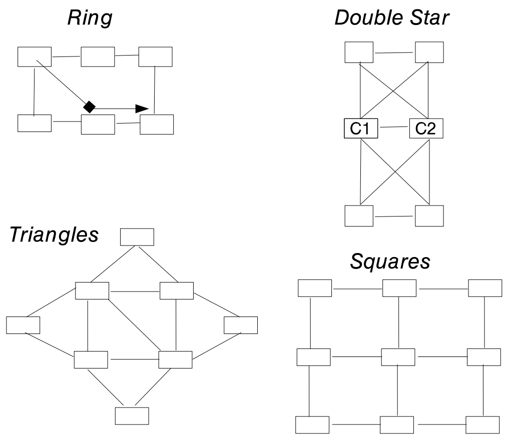


Figure 3.4: Regular topologies

The third topology, shown in the lower left part of figure 3.4, is a network composed of *triangles*. Triangles are commonly used as building blocks in large network topologies. In this network, when the IGP weights are set to one, all links are protected by using a loop-free alternate. In fact, when three routers, A,B and C form a triangle, then link $A \rightarrow B$ can always be protected by using a LFA if $Cost(C \rightarrow B) < Cost(C \rightarrow A) + Cost(A \rightarrow B)$. Otherwise, this link can be protected by using U-turn ($A \rightarrow C \rightarrow B$).

The fourth topology that we consider is shown in the lower right part of figure 3.4. In this network, the building block is a set of four routers arranged in a square topology. When the IGP weights of all links are set to one, all links can be protected by using U-turns. In fact, if $X_1 \rightarrow Y_1$ is an edge of a square in the topology, then a U-Turn exists if the metric of the link $Y_2 \rightarrow Y_1$, is lower than the metric of $Y_2 \rightarrow X_2$ plus $X_2 \rightarrow X_1$.

3.3.2 Real networks

To evaluate the network coverage of the IP-based fast reroute techniques, we considered five distinct ISP topologies : Abilene, GEANT and three commercial ISP networks. Abilene is a research network deployed over the continental US. It is composed of 11 routers and 14 (28 directed) links. The IGP weights on this network were apparently set according to the link delays. The topology of the Abilene network is shown in figure 3.5.

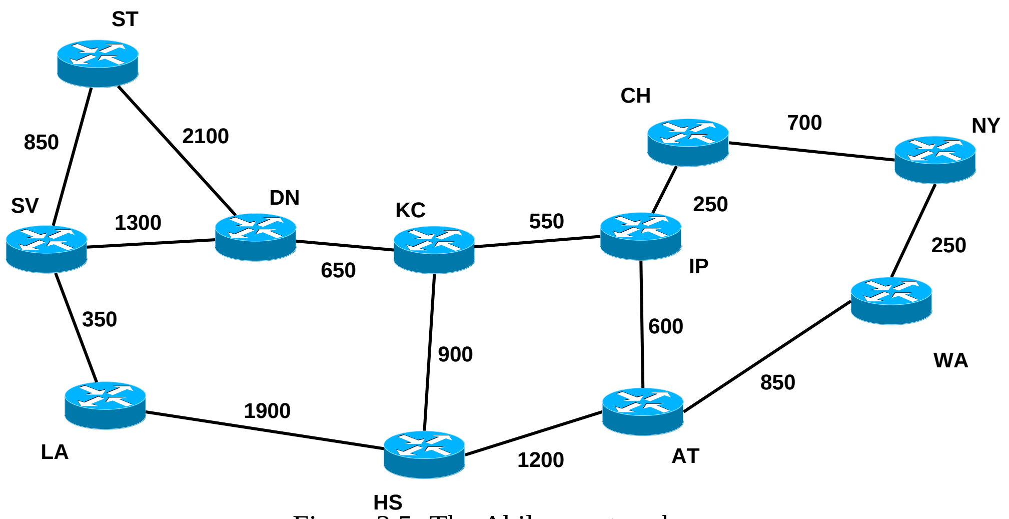


Figure 3.5: The Abilene network

GEANT is the European research network that links all National Research Networks in Europe together and to other research networks such as Abilene. GEANT is composed of 22 routers and 72 directed links. There is basically one router per European country and an additional one in New-York notably for the US peerings. Inside GEANT, the IGP weights were mainly set according to the measured link delays, with some manual tuning.

ISP1 is a commercial network covering an European country. The core of this network is composed of 190 directed links (64 directed links are backup links) and 50 routers. The setting of the IGP weights is mainly a function of the link bandwidth and favours high-speed links. Note that no protection must be provided to backup links when all the other links are up.

ISP2 is also a commercial network in an European country. The core of this network is composed of 11 routers and 26 links. Most of the IGP weights were set to 1 except for some manual tuning.

ISP3 is a Tier-1 ISP whose core is composed of 83 routers and 286 directed links. Due to the setting of the IGP weights, 21 directed links do not carry traffic and one link is only used in one direction. In this network, the setting of the IGP weights was tuned to meet some specific traffic requirements.

For the analysis of those networks, we removed from each topology the routers that were connected via a single link to the network. Clearly, the links of these routers cannot be protected since their failure partition the network.

It is important to note the differences in design between the research and the commercial networks under study. The two research networks are composed of a small number of Point of Presence (PoPs) to which their customers and peers are connected. Most PoPs of both networks contained a single router and there is a single high bandwidth link between two PoPs. In those research networks, the failure of one link or one router may cause a lot of traffic to be rerouted. Commercial networks are usually much more redundant. First, each PoP is usually composed of at least two core routers and possibly several aggregation routers to aggregate the traffic received from peers and customers [GM03, Gil05, Sie02]. Figure 3.6 shows two typical configurations. In the left configurations, the aggregation routers (AR) have a primary and a secondary connection to the core routers (CR). In this design, the IGP weights are set such that the secondary connections are only used if the primary link or core router fails. A second possible setting of the IGP weights is to use the same IGP weights for the links between the aggregation routers and the core routers to favour load balancing with ECM.

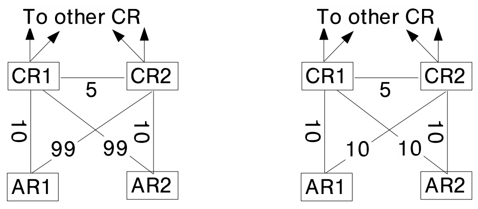


Figure 3.6: Typical PoP designs in commercial ISP networks

The PoPs are interconnected in different ways depending on the underlying physical infrastructure. Usually, each core router in a PoP is connected to at least two core routers in different PoPs in the network. A large PoP could of course be connected to more than two distant PoP. A typical interconnection between PoPs is shown in figure 3.7.

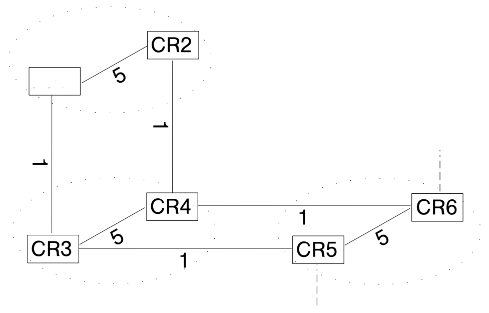


Figure 3.7: Typical PoP interconnections in commercial ISP networks

Network	Total number of Links	No LFA	1 LFA	2 or + LFAs
Abilene	28	16	10	2
GEANT	72	24	30	18
ISP1	114	52	38	24
ISP2	26	22	4	0
ISP3	265	92	116	57

Table 3.4: Coverage of loop-free alternates

We choose to limit our analysis of large networks to real network topologies because the settings of the IGP weights has a strong impact on the results. Using randomly generated network topologies such as those produced by BRITE [MAMB01] or GT-ITM [CDZ97] would not reproduce the redundancy and the IGP weights of real networks. For the same reason, we did not use the ISP topologies inferred by the Rocketfuel project [SMW02]. Since those topologies were inferred by using `traceroute`, they mainly contain the primary paths and rarely the secondary ones [TMSV03]. They cannot thus be used to evaluate protection techniques.

3.3.3 Simulation results

In this section, we present and discuss the network coverage of the different protection techniques. As described earlier, the simulator tests the simpler techniques first and only tries to use the more complex techniques when the simple techniques do not suffice.

Our first simulation results shown in table 3.4 consider the loop-free alternates. For each network topology, we provide the total number of (directed) links, the number of links for which there is no LFA and the number of candidate LFAs for the protectable links. Having several candidate LFAs to protect one link is interesting as this gives more choice to the protecting router.

Our simulations show that loop-free alternates is an effective protection technique for GEANT, ISP1 and ISP3. For Abilene, only 42% of the links can be protected by using loop-free alternates. The links of Abilene that can be protected with more than one single loop-free alternate are $HS \rightarrow KC$ and $AT \rightarrow IP$. Indeed, HS can protect $HS \rightarrow KC$ by sending packets towards either LA or AT, and WA and HS are loop-free alternates for link $AT \rightarrow IP$. Among the 16 links that cannot be protected by a loop-free alternates, a typical example is $KC \rightarrow DN$. This link cannot be protected as both HS and IP use it to reach DN. This network is shaped as 3 main rings which do not favour loop-free alternates.

The good network coverage in GEANT is because the network is basically divided in two parts. The first part is a highly meshed core and the second part a set of 9 distant POPs that are attached to 2 core routers. Each of the 18 links between a distant POP and a core router is protectable by a single loop-free alternate. Most

Network	Links not protected by LFA	No U-turn	1 U-turn	2+ U-turns
Abilene	16	4	9	3
GEANT	24	5	2	17
ISP1	52	32	6	14
ISP2	22	15	2	5
ISP3	92	13	37	42

Table 3.5: Coverage of the U-turns

of the links that can be protected with more than one LFAs are links inside the core. As the core of the network is highly meshed, there are many triangles with protectable links. The links were no LFA can be found are links from the core routers to the access routers and the links of the large ring providing connectivity to the Eastern Europe. The main reason for the low network coverage of loop-free alternates in ISP2 is that the core of this network is based on rings.

In table 3.5, we present the U-Turn coverage for the links that cannot be protected by using a LFA. The simulations show that the U-turns are useful in most topologies, except ISP1 and ISP2. In Abilene, 4 links (15%) are still unprotected. For example, $NY \rightarrow WA$ cannot be protected by using a U-Turn. Indeed, IP is the only neighbour of NY's neighbours. Unfortunately, it uses link $NY \rightarrow WA$ in its SPT. $CH \rightarrow IP$ is also unprotected as WA uses this link to reach IP. Potential U-Turns for $SV \rightarrow LA$ are DN and KC. But both have the link to be protected in their SPT. The last link that remains unprotected is $DN \rightarrow KC$.

The 32 directed links of ISP1 that cannot be protected with a U-Turn are links from core routers to access routers. All the routers within a 2-hop distance from those core routers utilize the link to be protected, so that no U-Turn can be found. We will see above that, in fact, all the routers of the core utilize these links, so that a directed forwarding tunnel will have to be used to protect those links.

The large rings that appear in ISP2 explain the absence of U-Turn Protection for 15 links.

We can see that 95 % of the links of ISP3 can be protected by using a LFA or a U-Turn. There only remains 13 out of the 265 links of the topology that require a protection tunnel. This is due to the good meshing of this network.

We summarise the network coverage of the combined LFA and U-Turns in the first columns of table 3.8. We can see that most of the links can be protected by sending packets to a neighbour or to a neighbour's neighbour.

We now consider the utilisation of protection tunnels to protect the remaining links. We can see in table 3.6 that all the links of GEANT and ISP2 are now protected. This means that in those topologies, for any link $X \rightarrow Y$, there is always at least one node Z such that $X \rightarrow Y \notin SPT(Z)$ and $X \rightarrow Y \notin Paths(X, Z)$.

In Abilene, links $CH \rightarrow IP$ and $DN \rightarrow KC$ cannot be protected. Indeed, all the routers (NY and WA) that CH reaches without using its link with IP have

Network	Links not protected with LFA/U-TURN	No TEP	1 TEP	2 or + TEPs
Abilene	4	2	1	1
GEANT	5	0	1	4
ISP1	32	32	0	0
ISP2	15	0	8	7
ISP3	13	8	0	5

Table 3.6: Coverage of protection tunnels

Network	Links not protected by LFA/U-TURN/TEP	No TEP+	1 TEP+	2 or + TEP+s
Abilene	2	0	2	0
GEANT	-	-	-	-
ISP1	32	0	29	3
ISP2	-	-	-	-
ISP3	8	0	6	2

Table 3.7: Coverage of protection tunnels with directed forwarding

$CH \rightarrow IP$ in their SPT. And all the routers that DN reaches without using its link with KC (SV, ST, LA) have $DN \rightarrow KC$ in their SPT.

Let us look at the protection tunnels that can be established for the link $SV \rightarrow LA$, to recover the reachability of LA and HS. SV can deviate its packets in a tunnel towards WA or AT. As the simulator chooses the closest tunnel endpoint, SV will select the tunnel towards AT and the length of the path to the protected destinations will be minimised. However, we can see that in this particular case, a tunnel with directed forwarding from SV to $KC \rightarrow HS$ would have given optimal paths to the protected destinations. We can also notice that using *NotVia* addresses in this case would have forced the packets from SV to HS to go to LA, which would have then forwarded them back to HS.

Note that this kind of unfortunate situations rarely occurs in sufficiently meshed ISPs. Even if Abilene is a small ISP, it is composed of three large rings that do not favour protection techniques.

However, protection tunnels did not help to increase the protection coverage for ISP1. ISP1 is basically a network with a small set of core routers and many distant POPs. The 32 unprotected links in this topology are the links between a core router and a distant POP. Those distant POPs are connected to two core routers. Unfortunately, for each concerned link $Core1 \rightarrow POP1$, the link $Core2 \rightarrow POP1$ is a backup link with a lower bandwidth and a higher IGP weight. Furthermore, router $Core1$ is never a neighbour of $Core2$. This implies that a U-Turn protection $Core1 - Core2 - POP1$ is impossible. All these links will thus have to be protected by using protection tunnels with directed forwarding.

In 3.7 we finally see that all the links can be protected by using protection

Network	Links	LFA	U-turns	Tunnel	Directed Tunnel	Notvia
Abilene	28	42%	85%	92%	100%	-
GEANT	72	66%	93%	100%	-	-
ISP1	114	54%	71%	71%	100%	-
ISP2	26	15%	42%	100%	-	-
ISP3	265	65%	95%	96%	100%	-

Table 3.8: Combined coverage of loop-free alternates, protection tunnels and NotVia addresses

tunnels with directed forwarding are enabled.

In Abilene, a protection tunnel with directed forwarding is necessary to protect link $DN \rightarrow KC$. The tunnel is $DN - SV - LA \rightarrow HS$. For link $CH \rightarrow IP$, the directed forwarding tunnel is $CH - NY - WA \rightarrow AT$.

Table 3.8 summarises the coverage of the IP-based fast recovery techniques in the studied network topologies. It shows clearly that by combining loop-free alternates, U-turns and protection tunnels, it is possible to protect all links in real ISP topologies. The values describe the percentage of links that can be protected by combining the first protection techniques. For example, in GEANT all links are protected by using LFA, U-turns and protection tunnels, while in Abilene protection tunnels with directed forwarding are required in addition to the techniques used in GEANT. The *not-via* addresses were not necessary to protect unicast IP traffic in the topologies that we considered.

3.4 Stability of the IP-based fast reroute techniques

Another issue to be considered with fast-reroute techniques is the stability of those protections when the network topology changes. Measurement studies of the link failures in large ISP networks [MIB$^+$04, WJL03, SIG$^+$02] have shown that their topology changes very often. After each topological change, the routers must recompute their SPT and update their FIB. As the IP-based protections also depend on the properties of the network topology such as the SPT, a link that is protectable by using a loop-free alternate at time t may be not protectable anymore by using this technique at time $t + 1$ after a distant link failure.

To analyse the stability of the protection techniques, we used our simulator to evaluate the impact of all possible individual links failures on the link protection established in each topology. For each directed link, we record the type of protection used and the neighbour in the case of loop-free alternates or U-turns and the tunnel endpoint when protection tunnels are used. For each topological change, we count the number of protections that are affected by the change. A topological change can affect a loop-free alternate or a U-turn by either forcing the protection router to select another alternate or forcing the protection router to use another technique. A topological change can affect a protection tunnel by forcing the protection router to select another tunnel endpoint or use a tunnel with directed forwarding instead of a normal tunnel or switch to not-via addresses.

As the current alternative to IP-based protection techniques is to utilize MPLS, we also plot, as points, for comparison purposes, the number of link disjoints MPLS tunnels that are affected by each topological change. Our simulator selects the link disjoint path with the shortest IGP weight to establish the MPLS tunnel. Trivially, such a link disjoint protection tunnel is affected by a topological change if the topological change was the suppression of any of the links on its path.

Note that after a topological change, some routers may become single-homed. For example, consider the Abilene topology shown in figure 3.5. In this topology, many routers have only two links. If link $AT - WA$ is removed, then none of the links attached to WA, NY and CH can be protected anymore. This is true for both the IP-based protection techniques and the utilization of link disjoint MPLS tunnels. Our simulator detects such cases and does not try to protect them by using either technique.

In figure 3.8, the curve shows the number of protections that must change after each topological changes in the Abilene network. We consider the 14 topological changes that correspond each to the failure of one link in the Abilene network. The topological changes were ordered in increasing number of affected protections. The topological change with the largest impact is the failure of $AT - IP$. This link is the intersection between the $HS - KC - IP - AT$ ring and the $IP - AT - WA - NY - CH$ ring. A link disjoint MPLS tunnel established to protect any of the links in those rings always uses link $AT - IP$. For the IP-based protection techniques, consider for example link $KC \rightarrow HS$. Before the topological change, this link was protected by using the U-turn $IP \rightarrow AT$. After the change, it is protected by U-turn $DN \rightarrow SV$.

Figure 3.9 provides the stability of the protection techniques in GEANT. The topological change with the highest impact is the failure of the most central link of the network. This link carries a large number of end-to-end paths. This link is used by many link disjoint MPLS tunnels. This explains why 30 MPLS tunnels are affected by this topological change. The main reason why there are fewer protection changes with the IP-based techniques compared to the link disjoint MPLS tunnels is that when there exists a loop-free alternate, it rarely changes when a distant link is removed. On the contrary many link disjoint MPLS tunnels are long. Thus, they are affected by more topological changes.

Finally, we provide in figures 3.10 and 3.11 the stability of the protection techniques in ISP1 and ISP3. When considering the 143 topological changes in ISP3, we found that the link disjoint MPLS tunnels are less affected than the IP-based techniques for 31 changes, while the IP-based techniques are less affected for 55 changes. For the other 57 changes, as many link disjoint MPLS tunnels as IP-based protections are affected by the change. This means that in ISP3 the IP-based protection techniques are slightly more stable than the link disjoint MPLS tunnels. A similar result is found when considering the stability of the protection techniques

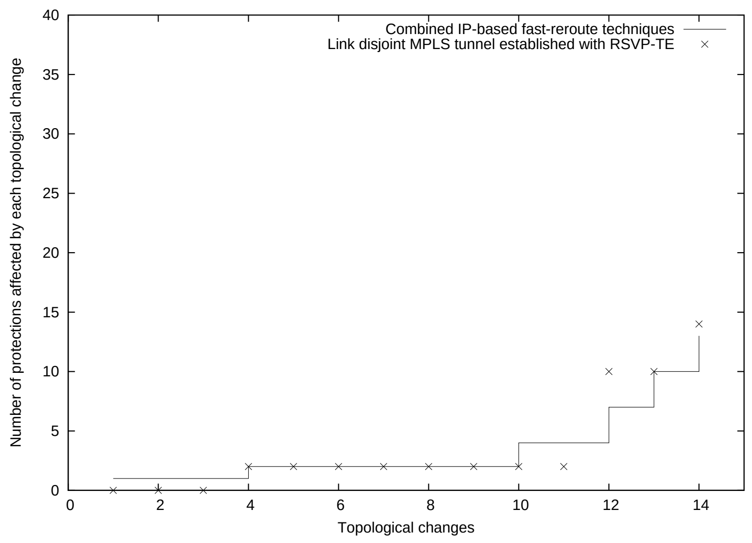


Figure 3.8: Stability of the protection techniques in Abilene

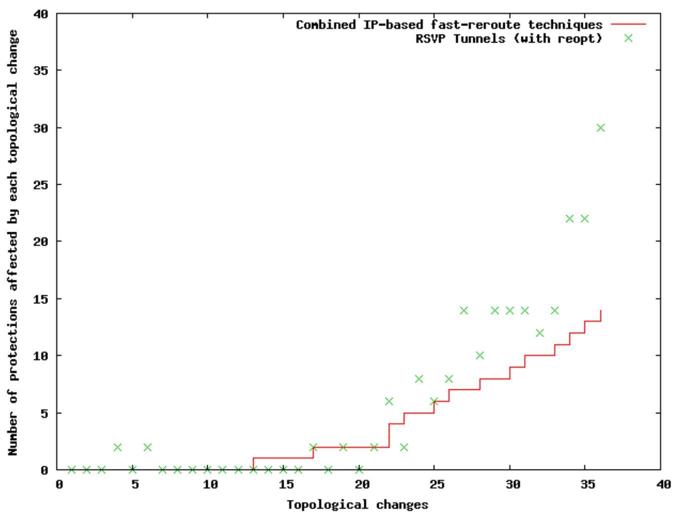


Figure 3.9: Stability of the protection techniques in GEANT

in ISP1.

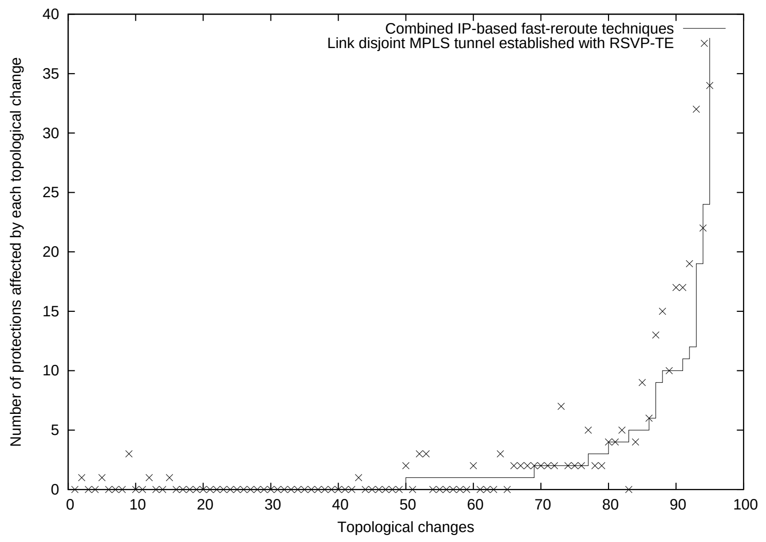


Figure 3.10: Stability of the protection techniques in ISP1

3.5 Applicability of IP Fast Reroute techniques

In this section, we review the applicability issues of the various techniques analysed in this chapter. We firstly present the inherent issues of some of the techniques which may prevent them to be deployed. Next, we discuss the applicability of IP Fast Reroute techniques for the protection of LDP traffic.

3.5.1 Issues with IP Fast Reroute techniques

Relying on ECMP to achieve fast restoration in case of failure is very impractical. An ISP that relies on such a technique has to design its topology to have Equal Cost Shortest Paths on all source-destination paths. This technique is used by some ISPs for some important links, but protecting all the links of a topology using ECMP would be too expensive. Indeed, to take advantage of ECMP repairs, the topology must be designed by respecting the constraint of having equal cost paths from each source to each destination. This must be combined with traffic requirements inside the network, which can lead to complicated network designs. Furthermore, some ISPs engineer their link metrics to avoid ECMP, this for practical, troubleshooting purposes. In those cases, ECMP protection cannot apply.

The main drawback of the U-Turn technique is that it requires modifications to the forwarding performed on the routers interfaces. It requires to mark packets following alternate paths in order to let the receiving router find out that the packets

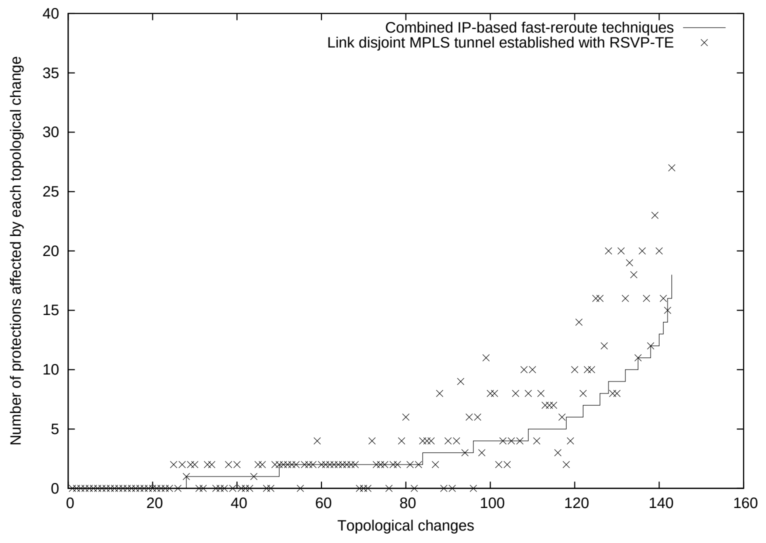


Figure 3.11: Stability of the protection techniques in ISP3

it receives must not be forwarded according to its regular FIB, but using its U-Turn FIB. Also, it requires to have distinct U-Turn states in each interface FIB. When put in balance with the provided coverage gain, this technique no longer looks attractive.

The main issue of protection tunnels is that the end-to-end path for source-destination pairs recovered by using them can be hazardous. For example, such kind of protection technique can let an intra US link be protected with a tunnel whose endpoint is in Japan, although an intra US recovery paths is feasible to protect this link.

Due to the automatic and self healing nature of IP FRR schemes, such protection tunnels would have to be continuously monitored by the operator. Indeed, upon a topological change, the operator would have to check if the routers do not start considering hazardous paths to protect their links, and prevent the routers from using them if required.

Although the coverage of this technique was very good in our coverage analysis, even for ISPs with asymmetrical link metrics, the end-to-end path that they use to achieve such a good coverage turns to be a show stopper for the deployement of IP protection tunnels as defined in [BFPS05].

At the time of this writing, these issues motivated the IETF RTG working group to adopt an "LFA + NotVia" as the recommended IP Fast Reroute suite.

3.5.2 IP Fast Reroute techniques and LDP traffic

Currently, ISPs using MPLS tunnels established with the Label Distribution Protocol (LDP) [ADF$^+$01] to forward IP traffic across their network have to enable

MPLS Resource ReSerVation Protocol (RSVP) [BEZ+97] on each of their links for the simple purpose of Fast Reroute. In order to protect their traffic from link failures, a single hop MPLS-TE tunnel is established on each link of the network, and an MPLS FRR tunnel is established to protect each of these single hop tunnels.

An attractive feature for IP Fast Reroute would be its ability to also protect LDP traffic, so that such ISPs would benefit from it.

The main question to be answered when protecting LDP traffic is to preserve the validity of the labels used to forward the protected traffic. With MPLS, the significance of a label is local to a link, so that when a packet is deviated on an alternate path, care must be taken to swap the labels accordingly.

In the following sections, we discuss the applicability of IP Fast Reroute to protect LDP traffic. We first discuss FIB organizations issues bounded with LDP protection. Then, we consider the ability of each Fast Reroute technique to protect LDP traffic. Finally, we discuss which LDP label distribution modes are recommended to facilitate the deployement of IP FRR.

Organization of the FIB

To emphasize the constraints on the FIB organization implied by the protection of LDP traffic, let us consider the protection of link $N \rightarrow E$ in figure 3.12.

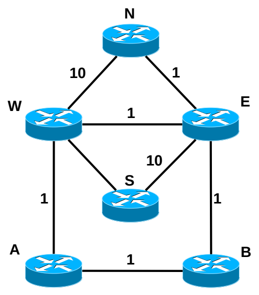


Figure 3.12: Simple network topology

N forwards packets destined to A and B via link $N \rightarrow E$. When LDP is used, N received from E the labels that must be used to forward packets to these nodes. Let us assume that the label mappings are such that N must use label l_{A_E} to send packets towards A along link $N \rightarrow E$ and label l_{B_E} to send packets towards B along link $N \rightarrow E$. W can be used by N as a Loop Free Alternate for such destinations when $N \leftrightarrow E$ fails. However, the label to be used to forward packets towards A along $N \rightarrow W$ is not necessarily equal to l_{A_E}. Indeed, a different label, say l_{A_W} might have been advertised by W, and this label must be used to let the deviated packets be appropriately forwarded by W.

That implies that it is not sufficient for N to know the link over which it can deviate packets towards a given Forwarding Equivalence Class (FEC). It must also maintain which label must be used to send these packets along that link. As this information differs for all the FECs reached via a given link, the most appropriate FIB organization to protect LDP traffic is similar to the one illustrated in the bottom part of figure 3.1. For each FEC or LDP label in the FIB, there must be a primary (oif,label) pair to be used as well as a secondary (oif,label) pair to be used.

In the remainder of this section, we will discuss the applicability of the various IP FRR techniques to protect LDP traffic. We assume that the FIB organization proposed above is being used.

Applicability of ECMP to protect LDP traffic

When using ECMP protections, the protecting node N will deviate packets towards a given FEC to a neighbor B that it was already using to reach this FEC. Thus, there are no specific issues to use ECMP protection in the context of LDP traffic, as the node already received the required label mapping to forward such packets over alternate links. In figure 3.13, node N will protect the reachability of FEC f from the failure of link $N \rightarrow A$ by deviating the packets along $N \rightarrow B$. As this node already has the label mapping $(f, L2)$ to reach f via this link, nothing else has to be modified in N.

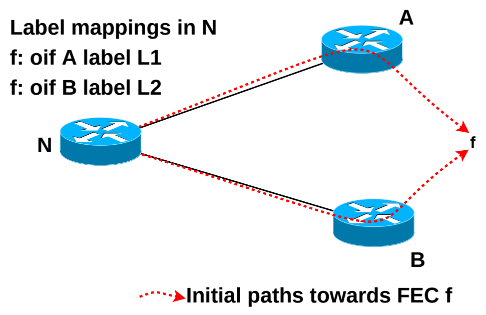


Figure 3.13: ECMP repair for LDP

Applicability of LFAs to protect LDP traffic

When using LFAs, a protecting node N will deviate packets towards a FEC f to a neighbor B that N is not already using to reach this FEC. To be able to quickly reroute packets towards B, N must know in advance which labels must be used to forward packets along link $N \rightarrow B$.

So, once an LFA has been found to protect a FEC f, the protecting node must obtain a label mapping for f from this LFA.

In figure 3.14, node B is an LFA of node N, protecting link $N \rightarrow A$. In order to know which label must be used when the LFA is used, N must send an LDP

Label Request message to B[1].

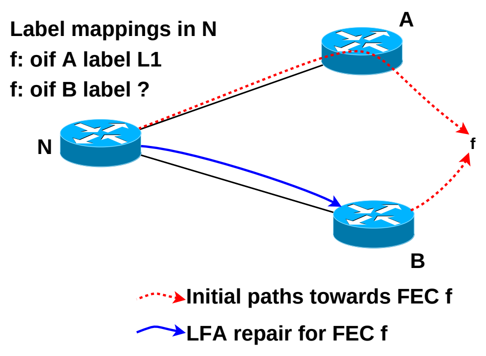


Figure 3.14: LFA repair for LDP

Applicability of U-Turns to protect LDP traffic

The same requirements as for LFAs apply in the case of U-Turns. The protecting node N must know the label to be used to deviate packets to a neighbor B. If B has to forward such packets to a neighbor C different from the neighbor that B normally uses to reach the corresponding FEC, B must also request a label for that FEC from C.

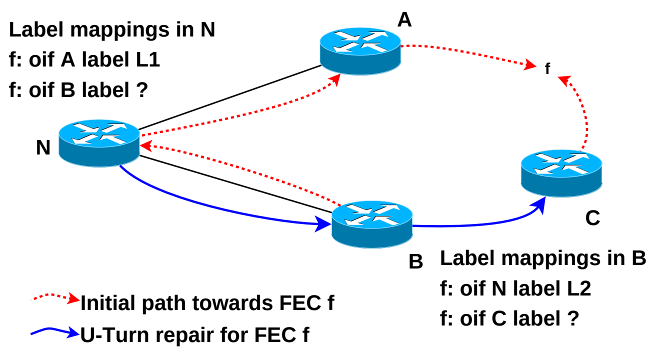


Figure 3.15: U-turn repair for LDP

[1]Note that this is only true if the “downstream on demand” label distribution mode is used. More details are provided on this mode latter in this chapter.

Applicability of Protection Tunnels for LDP traffic

Protecting LDP traffic with Protection tunnels is not an easy task. With this technique, the protecting node N deviates traffic inside a protection tunnel towards an endpoint D that is not affected by the failure. A packet being forwarded on that tunnel will be encapsulated using two labels. The outer label is used to reach the tunnel endpoint D and the inner label identifies the FEC corresponding to that packet, e.g its egress point. Once the packet is pushed in the tunnel, the inner label will not be used or switched until the packet reaches D. As N must set this label consistently with the LFIB of D, it must establish a remote LDP peering session between N and D, so that N can set the appropriate inner label. Let us consider this issue in figure 3.16. N wants to protect link $N \rightarrow A$ with a protection tunnel whose tail-end is D. Initially, N was sending packets towards f to A, encapsulated using label $L1$. Upon failure detection, N will tunnel the packets to D. To do that, it will send them to B, using a second encapsulation with label $L2$. However, N cannot use label $L1$ as inner label as this label is not mapped with f in D. Thus, N must have an LDP session established with D, so that it can learn the label mapping $(f, L3)$ from D. The packet towards f forwarded by N along the protection tunnel will thus have an outer label $L2$ so that it will be correctly switched to D, and an inner label $L3$ so that D will correctly switch it towards f.

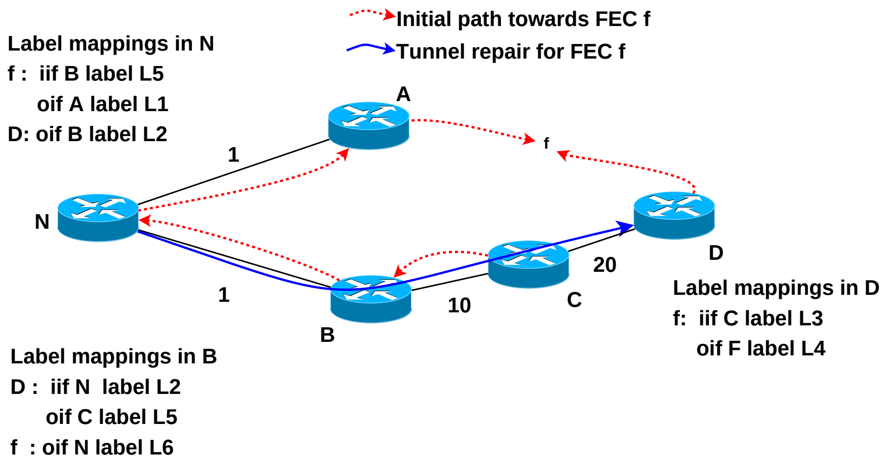


Figure 3.16: Tunnel repair for LDP

When using Protection Tunnels with direct forwarding, the problem is even more complex. N must use an outer label to reach D, an inner label to identify the FEC. But, also the neighbor to which D will send the packet must be made identifiable by D. To do that, multiple FEC, and thus label mappings would have to be originated by D. Each FEC would identify the link that D must use when receiving packets encapsulated in the label corresponding to this FEC.

Applicability of NotVia to protect LDP traffic

When using NotVia, in link protection mode, the protecting node N will encapsulate an LDP packet in another LDP packet corresponding to the NotVia address used for the protection. To do this, it must be ensured that the tail-end of the tunnel will correctly switch the LDP packets received over this tunnel. The solution to this issue depends on the label space used for distribution by the tunnel tail-end.

If the tunnel tail-end uses a **per-platform label space**, N can use the same inner label as it would use to forward the packet over the protected link. Indeed, the tail-end of the link is also the tail-end of the protected link, so that the packet received over the tunnel can be forwarded as is.

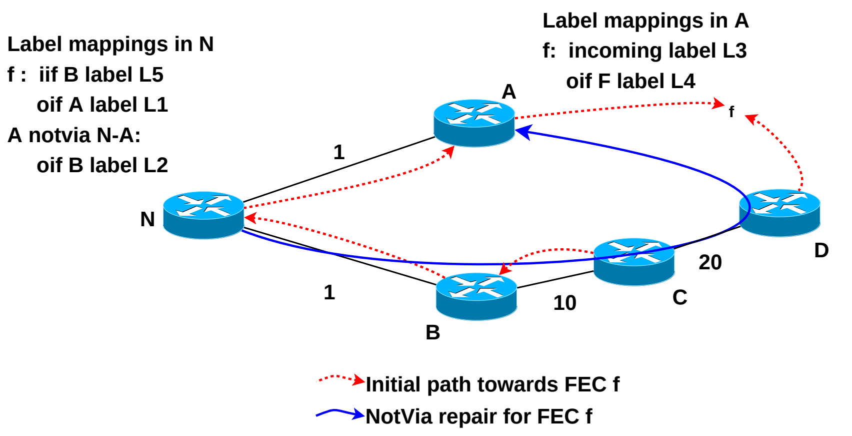


Figure 3.17: NotVia repair for LDP

This case is illustrated in figure 3.17. N protects link $N \rightarrow A$ using NotVia address $A_{N\rightarrow A}$. N received the label mapping for FEC f over link $N \leftrightarrow A$, so that it knows that A's incoming label for FEC f is $L3$. To deviate traffic destined to f when $N \rightarrow A$ fails, N will swap the packets towards f that it receives, using label $L3$, and encapsulate this packet in another LDP packet with label $L2$, in order to forward it to B. When the packet reaches A, A pops the label and forwards an LDP packet whose incoming label is $L3$, so that it knows it has to swap it to $L4$ and forward it towards f.

If the tunnel tail-end uses a **per-interface label space**, the inner label that must be used by N differs according to the interface at which the tunnel ends. Indeed, the label switching of packets received over an interface depends on the interface itself. In that case, the tunnel tail-end must send the label mappings for each protected FEC to N in order to have N switch the inner labels so that they match those used on the incoming interface terminating the protection tunnel. It is therefore necessary for the tunnel end-point to know which interface terminates the protection tunnel.

This case is illustrated in figure 3.18. In this case, the incoming label for pack-

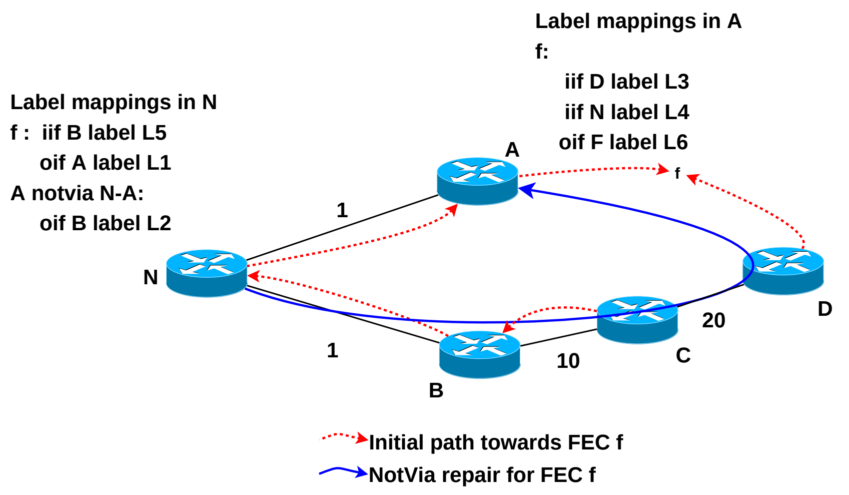


Figure 3.18: NotVia repair for LDP with per-interface label space

ets towards FEC f is $L3$ when the incoming interface is the one connected to $A \leftrightarrow D$ and $L4$ when the incoming interface is the one connected to $A \leftrightarrow N$. Thus, N does not have the sufficient information to appropriately swap the label of packets destined to f when the NotVia repair has to be used. In order to solve this issue, a new LDP request message should be used to let N retrieve label mapping information bound with the interface of A connected to $A \leftrightarrow D$. Another mean could be to let A establish a remote LDP peering session with D to let A retrieve such information with regular label request message. This last solution is only feasible when downstream label distribution and liberal retention modes are used by D. If it is not the case, D will not know the label to be used when it does not use the link $D \rightarrow A$ to reach f.

IPFRR to protect LDP traffic and label distribution modes

Label advertisement can operate in either "downstream on demand" or "downstream unsolicited" modes [ADF$^+$01]. Under downstream on demand mode, an LSR only sends a label mapping upstream as a response to an explicit label request from the upstream node. Under downstream unsolicited mode, an LSR sends label mappings upstream even for the FECs that were not requested by the upstream node.

Label retention mode can operate in either "liberal" or "conservative" modes [ADF$^+$01]. Under liberal mode, an LSR retains a mapping received from a downstream node even if this LSR does not use the downstream node to forward packets to the corresponding FEC. Under conservative mode, only the mappings actually used for forwarding are retained by the LSR.

When IPFRR is used in conjunction with LDP, routers will have to forward deviated packets over links that are not initially used to reach some FECs in the network. Under conservative retention and downstream on demand modes, the lack of label information over these links will dramatically slow the restoration, as the recovery will only take place after an exchange of label mappings among the routers. In order to guarantee a fast recovery, downstream unsolicited advertisement and liberal retention modes should be used.

3.6 Related work

One of the first approach to provide fast-reroute in IP networks was described in [NST99]. Narvaez et al. developed local restoration techniques to provide restoration in case of link failures. The first technique discussed in [NST99] is the utilization of tunnels. It was rejected in 1999 based on the inefficiency of the packet encapsulation and decapsulation on high-end routers at that time. Today, thanks to the need to support VPN services, recent routers can perform encapsulation and decapsulation at line rate. Then, [NST99] proposed a technique to allow the protection router to inform the routers on the restoration path that they should update their routing table. This avoids the need to flood a new link-state packet throughout the network and provides faster convergence since only a few routers need to update their FIB. As this technique requires to update the FIB of several routers, it will achieve a longer restoration time than the techniques discussed here, where only the protection router needs to reroute.

In [LYN+04], Lee et al. proposed a local rerouting technique called failure insensitive routing (FIR). This approach prepares the failure and is thus similar in principle to the techniques discussed in this thesis. As it relies on per-interface specific forwarding tables, it belongs to the same family of solutions as the U-turn technique discussed in this part of the work. Another similar technique was proposed in [ZKN+05]. Note however that compared to the techniques described here, that quickly reroute the packets affected by a failure, the solution described in [ZKN+05] discards affected packets to avoid transient loops.

In [SCK+03], Schollmeier et al. proposed a new routing scheme called $O2$ that allows each router to compute several paths for each destination. As a protection router has alternate paths to reach all destinations it can easily react to any failure. However, the main drawback of this new routing scheme is that it cannot be used in current networks that are using OSPF or ISIS.

A few other fast reroute protection techniques have been proposed [Nai04b, Nai04a] but are not really considered within IETF.

3.7 Conclusion

To provide faster recovery in case of failures in IP networks, the IETF is currently investigating several fast-reroute techniques : loop-free alternates, U-turns, protection tunnels and NotVia addresses. With those techniques, a router that detects a local link failure can forward the packets affected by the failure to either directly an alternate neighbour or by encapsulating them inside a tunnel towards a distant router. We have explained those techniques as well as their advantages and drawbacks.

Compared to currently deployed fast-reroute techniques like MPLS-based link disjoint tunnels, a first concern with the IP-based fast reroute techniques is their coverage, i.e. the number of links that can be protected by each technique in large networks. We have obtained the real topology of five very different networks and shown by simulation that loop-free alternates combined with U-turns are sufficient to protect between 40 and 90% of the directed links. Furthermore, in all studied networks, adding protection tunnels to those two basic techniques was sufficient to achieve a full coverage. We also evaluated the stability of the IP-based fast reroute techniques by simulating the impact of distant link failures on established IP-based protections. Our simulations indicate that the IP-based techniques are as stable as the currently deployed MPLS-based link disjoint tunnels.

However, U-Turns and Protection tunnels suffer from drawbacks, which render them less attractive when compared to the gain in coverage that they provide. These issues tend to favour a combination of LFAs and NotVia addresses. This combination would provide a lightweight protection for the links whose surroundings allow their application, while only enabling the heavy machinery of NotVia to protect the few links that are not fully covered with LFAs.

Chapter 4

Transient Forwarding Loops during IGP convergence

During our study of the convergence time, presented in chapter 2, we found out that the recovery of the reachability was not always obtained once the routers adjacent to the failing ressources have updated their FIB. Indeed, when not all the routers have updated their FIB according to a given topological change, transient **FIB inconsistencies** and forwarding loops can occur among the routers. Such loops finally lead to packet loss because they are responsible of TTL expiration and link saturation.

IP Fast Reroute techniques do not solve these problems on their own. Indeed, even if a failed link was protected with a Fast Reroute technique, the adaptation of the routers to the new topology, considering the removal of the failed link, can lead to transient inconsistencies **upstream** of the failed link.

When a link has to be shutdown due to a **maintenance operation**, packets can also be lost. First, the command issued to the router can be processed abruptly, turning this predictable event into an event that is harmful for packet delivery. To reduce the impact of this issue, the operator can previously set the metric of the link being brought down to $MAX\_METRIC - 1$ [TR06], in order to have this link no longer belong to any shortest path accross the network when the shutdown is performed. However, transient inconsistencies can occur among the routers during this transition, and packets can still be lost.

The first objective of this chapter is to study the potentiality of transient forwarding loops in real topologies. The approach is different from the simulation study of chapter 2 as here we identify all the potential forwarding loops, independently of the flooding dynamics and FIB update times of the routers. We will see in this analysis that forwarding loops can occur for a significant number of topological changes in the network topologies under study. Also, we will observe **pathological cases** where a very large number of destination nodes can be affected by forwarding loops after a single failure. The results of this analysis motivate the design of loop avoidance schemes for the IGP.

The second objective of this chapter is to identify cases where a loop avoidance mechanism could be used to provide a totally loss free convergence.

This chapter is organized as follows. Section 4.1 presents the methodology used to identify potential forwarding loops. Section 4.2 presents the results of analysis performed on various network topologies. Next, section 4.3 discusses the impact of microloops on IP packet flows and section 4.4 introduces insights on when these loops could be avoided. Finally, section 4.5 draws some conclusion.

4.1 Capturing the transient forwarding loops in a network topology

This section describes the algorithm that we use to identify the forwarding loops that can occur upon a topological change. The algorithm answers the following question : "For a given link failure or link installation, what are the destinations for which there exists an ordering of the FIB updates that triggers a transient forwarding loop ?".

Due to the number of nodes in the topologies under study, it is not practical to generate all the potential orderings of the FIB updates and test if they lead to forwarding loops. Instead we will rely on theorem 4.1.1.

This theorem uses the notion of reverse Shortest Path Trees (rSPT). The rSPT of a node N in a weighted direct graph G, $rSPT(N, G)$, is the acyclic directed graph G made of the merging of the set of shortest paths from all the nodes in the network to N, in the graph G. It can be computed by carrying out the computation of the Dijkstra algorithm on the graph obtained by swapping the weights of each edge in the graph (namely the link metrics), the metric of link $X \rightarrow Y$ becoming the metric of the link $Y \rightarrow X$, and vice versa.

Theorem 4.1.1 *Given a link l, a destination d, and a network graph G, there exists an ordering of the FIB updates among the nodes of a network that leads to transient forwarding loops, when l is removed from (or added to) G, turning G into G', iif the merging of $rSPT(d, G)$ and $rSPT(d, G')$ contains a cycle.*

Proof :

The presence of a link $X \rightarrow Y$ in G means that X is forwarding packets destined to d to its neighbor Y, before having updated its FIB according to the topological change.

The presence of a link $X \rightarrow Y$ in G' means that X is forwarding packets destined to d to its neighbor Y, after having updated its FIB according to the topological change.

If there is a cycle in the merging of G and G', there exists a simple cycle C in this graph. As C is simple, it captures an instant during the convergence when some of the routers belonging to C have updated their FIB at this time, and some of them have not. If a packet destined to d reaches a member of C, it will be forwarded along this cycle, so that it is caught in a loop. ∎

Algorithm 1 presents a pseudo-code for the detection of destinations potentially affected by forwarding loops during the convergence following the failure of a link. Destinations potentially affected by such forwarding loops are also potentially affected by loops when the link comes back up to service. Note that we perform a trivial testing of the presence of length-2 cycles during the merging of rSPTs, in order to spare the unnecessary execution of a cycle detection algorithm on the obtained merging of the graphs. In other words, when there is a simple two-hop loop in the merging of two rSPTs, we do not carry out a full cycle detection.

```
Compute the set of nodes suffering from potential transient forwarding loops
in network graph $G$, upon failure of link $X \rightarrow Y$ :

Set LoopyDestinations = {};
graph SPT(X) = Dijkstra(G,X);
Set AffectedDestinations = nodes downstream of $X \rightarrow Y$ in $SPT(X)$;
Graph $G'$ = $G \backslash X \rightarrow Y$;
foreach $d \in$ AffectedDestinations do
    //Run dijkstra on $G$ with root $d$, with swapped weights
    Graph rSPT(d) = rDijkstra(G,d);
    //Run dijkstra on $G'$ with root $d$, with swapped weights
    Graph rSPT'(d) = rDijkstra(G',d);
    //Merge both acyclic graphs. Detect length-2 loops during the merging
    Boolean two-hop-loop = merge(rSPT(d),rSPT'(d),mergedRSPTs);
    if two-hop-loop then
        add(LoopyDestinations,d);
    end
    else
        //Detect longer cycles;
        if (detectCycles(mergedRSPTs)) then
            add(LoopyDestinations,d);
        end
    end
end
```

Algorithm 1: An algorithm to detect potential transient forwarding loops

4.2 Topology Analysis

In this section, we analyse the potential forwarding loops that can occur in 4 ISP topologies. We applied the Algorithm 1 to discover the potential forwarding loops that can occur when a link fails in those topologies.

In figure 4.1, we see the cumulative distribution of the percentage of destination nodes that are potentially affected by microloops upon the failure of each link of the topology. Figure 4.2 shows a zoom on the bottom part of this figure.

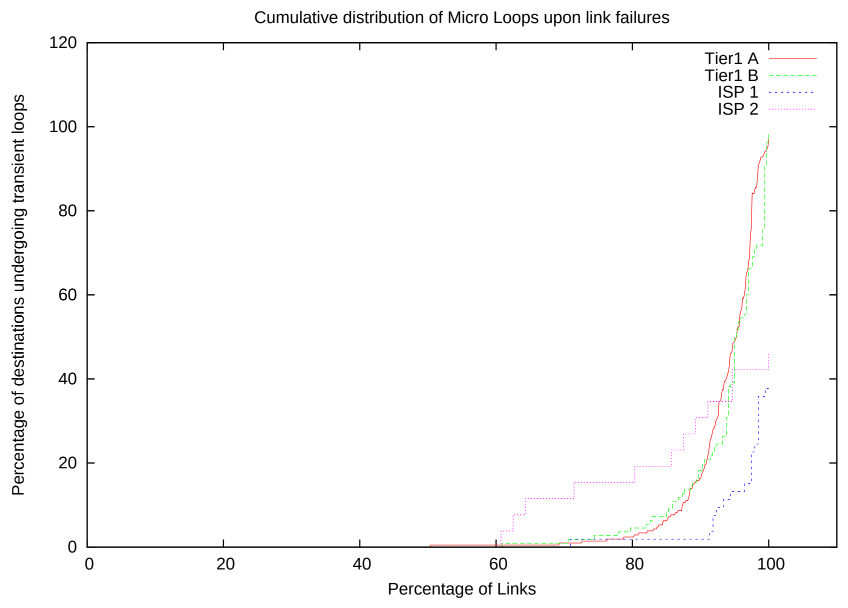


Figure 4.1: Destinations affected by micro loops upon link failures in 4 ISP topologies

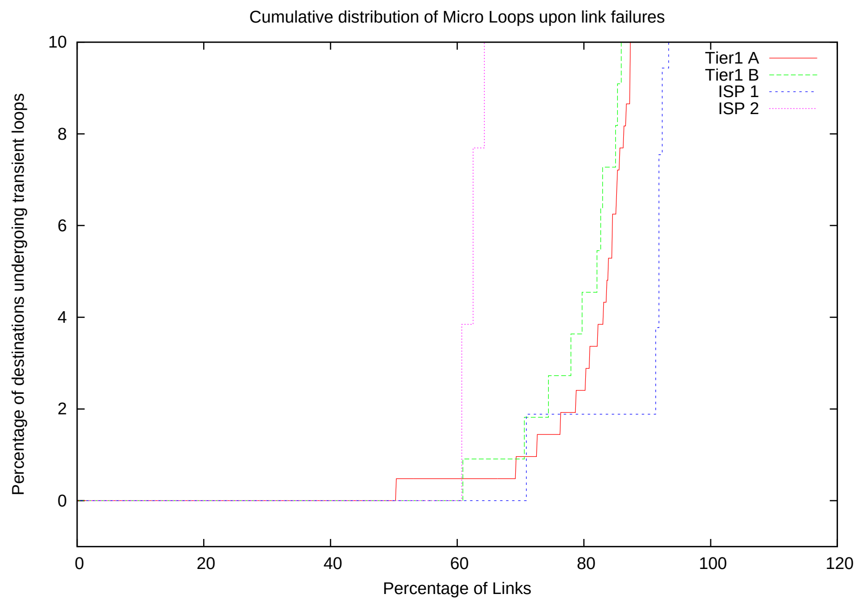


Figure 4.2: Destinations affected by micro loops upon link failures in 4 ISP topologies (Zoom)

Topology “Tier-1 A” is a large Tier-1 ISP containing approximately 200 nodes and directed 800 links. “Tier-1 B” contains approximately 110 nodes and 400 directed links. Topology “ISP 1” is a regional ISP containing around 50 nodes and 200 directed links and topology "ISP 2" is a small local ISP containing around 30 nodes and 60 directed links.

Our first observation is that the distribution of the percentage of destinations affected by micro loops upon a link failure is similar for the two Tier-1 ISPs.

We can see that forwarding loops are possible for 50% of the links of the topology in Tier-1 A. 20% of the links can lead to forwarding loops for only 1 destination node, typically for the tail-end of the failing link. For 17% of the links of the topology, i.e. approximately 130 links, the number of destinations affected by forwarding loops is larger than 10. 5% of the links are pathological, with a number of destinations affected by forwarding loops between 100 nodes and all the destination nodes excepting the head-end of the failed link. The topology “Tier-1 B” shows a very similar behaviour. Microloops are possible for 40% of the links of the topology. 10% of the links can lead to forwarding loops for only 1 destination node. For 15% of the links, the number of destinations affected by forwarding loops is larger than 10. The last 5% of the links are pathological, with a percentage of destinations potentially affected by microloops reaching up to very close than 100%.

Typically, the pathological cases are links in the PoP or links from the PoP to the core of the network. These links carry traffic from this PoP towards all the other nodes of the network. The design of both topologies looks similar in those places of the network, having pops designed as squares or rings, which have an important forwarding loop potential by essence.

The less pathological cases are actually the reverse direction of the links that can suffer from a large number of micro loops. These are the links carrying traffic from the core of the network towards each pop, hence the smaller number of affected destinations.

The shapes of the two smaller ISP topologies are similar to a double star. Though, these double stars are unbalanced, with access routers reaching a part of the destinations via one star, and the other part via the other. That is, one given link never carries the traffic from one pop towards all the other pops of the network. This explains why there are no such cases where nearly all the destinations can suffer from microloops upon a single link failure, as was observed in the Tier-1 ISPs. Still, around 40% of the links can lead to forwarding loops upon failure in both topologies.

4.3 Impact of micro loops

Let us analyse the impact of micro loops on a packet flow, based on Figure 4.3. In this topology, node A is sending packets towards node B, along the shortest path $A \rightarrow R \rightarrow X \rightarrow Y \rightarrow B$. Let us assume that the link between X and Y fails.

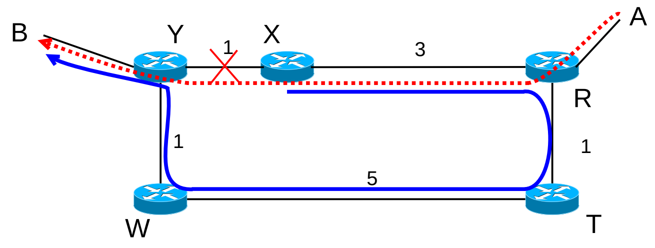


Figure 4.3: Example of micro loop

First, this flow will experience **packet loss** because X forwards traffic destined to B along a failed link. When X detects the failure of the link, as described in 2.3, X will originate a new Link-State packet and flood it accross the network. Then, X will recompute the shortest paths from X towards the other nodes of the network and update its FIB accordingly. The new shortest path from A to B after the failure is $A \rightarrow R \rightarrow T \rightarrow W \rightarrow Y \rightarrow B$. However, R will update its FIB after X, so that it is highly probable that packets destined to B are forwarded by R to X while X forwards them back to R, hence a forwarding loop $A \rightarrow R \rightarrow X \rightarrow R \rightarrow X$ ... occurs. When R has updated its FIB, the reachability may not be recovered yet. It is indeed very likely that T updates its FIB after R. So, when R deviates packets along $R \rightarrow T$, T will forward them back to R, hence a new forwarding loop $A \rightarrow R \rightarrow T \rightarrow R \rightarrow T$ ... occurs.

When the packets of a flow between a source and a destination are caught in a micro loop, packets will be **delayed** by the duration of the micro loop. In our example, routers R and T must have updated their FIB in order to have the packets forwarded by A towards B actually reach B.

Also, packets of this flow will be **re-ordered** as packets reaching one of the member of the loop upon the break of the loop will reach the destination before those that still travel along the loop. Let us illustrate this re-ordering with an example. Let us denote by p_i the ith packet to be forwarded by A towards B for the considered packet flow. At time t_0, R forwards p_1 to T. T has not updated its FIB yet, and forwards this packet back to R at time t_1. At time t_2, R receives p_2 from A, and forwards it to T. At time t_3, T updates its FIB for destination B. At time t_4, R receives p_1 from T and forwards it back to T. At time t_5, T receives p_2 forwards it along $T \rightarrow W$. At time t_6, T receives p_1 forwards it along $T \rightarrow W$. We clearly see with this example that packets can be re-ordered due to the forwarding loop.

Packets caught in a forwarding loop can be **dropped** if their Time-to-Live (TTL) reaches zero before the loop is broken by the routers. This issue can have a worsening effect on the convergence time. Indeed, when a packet TTL reaches zero, the receiving router may need to send an ICMP packet back to the source to notify the error to the source of the packet. Such a behaviour consumes CPU ressources on the interface from which the ICMP packet is sent. But, these ressources are critically required during the convergence, as these CPU are in charge of updating the FIB on the interface. Forwarding loops leading to ICMP packet generation can thus be the cause of a delaying of the FIB updates required to break forwarding loops.

Another source of packet drops is from bandwidth usage induced by forwarding loops on the links where the loop takes place. Let us assume that a forwarding loop of 50 msec on a link with a one-way delay of 5 msec. This scenario is realistic in nowadays routers and networks. If 20% of the link bandwidth was used before the event to reach destinations that are caught in the loop, the link will be **saturated** after less than 20 msec, and packets will be dropped due to this saturation for the remaining 30 msec.

4.4 When should micro loops be avoided

Several types of changes can occur inside the topology of an IP network. The most common type of change is the failure of a link [MIB+04]. A network typically contains point-to-point links and LANs. Point-to-point links are typically used between Points of Presence (POPs) while LANs are mainly used inside POPs.

We distinguish two different cases for link failures. First, if the failing link is not locally protected, the IGP should converge as quickly as possible. Second, if the link is protected with an IP Fast Reroute technique or another technique [ATC+04, BSP06], the IGP should converge without causing transient loops as the traffic passes through the tunnel during the IGP convergence.

It should be noted that link failures are often caused by manual operations and these can be considered as planned events. Surveys conducted by a large ISP [ICBD04] revealed that, over a five month period, 45 % of the failure events occurred during maintenance hours. Another ISP [DFM04] indicates that over one month, 75 % of the IS-IS events were caused by maintenance operations. Another study [MIB+04] mentions that 20 % of *all link down events* were planned. Those planned events should not cause transient forwarding loops [DFM04]. In the case of a maintenance of a link, some operators set the metric of the link to MAX_METRIC in order to let packets be forwarded on the link during the convergence [TR06]. However, doing this is not sufficient as transient loops can still occur .

It is also important to consider the increasing integration between the IP network and the underlying optical network [BRS03]. As the integration with the optical layer increases, the topology of IP networks will change more frequently than today. For example, [PDRG02] proposed to allow routers to dynamically establish optical links to handle traffic spikes. Similar approaches have been proposed with MPLS tunnels. Once a new optical link or MPLS tunnels becomes active, an IGP adjacency will be established between the attached router and the link will be advertised in the IGP [SS04]. Unfortunately, the addition and removal of each of those tunnels can cause transient loops in the network.

Another source of changes in IP networks are the IGP metrics. Today, network operators often change IGP metrics manually to reroute some traffic in case of sudden traffic increase [TR06]. Furthermore, several algorithms have also been proposed to automate this tuning of the IGP metrics for traffic engineering purposes [FRT02]. Today, those algorithms are mainly implemented in network planning and management tools [FGL$^+$00, BLD$^+$07]. However, ISPs are still reluctant to use such tools to frequently change their IGP metrics as each change may create transient forwarding loops in their network.

A second type of important events are those that affect routers. Routers can fail abruptly, but often routers need to be rebooted for software upgrades. For example, figure 6 of [MIB$^+$04] shows that during September and October 2002, many links of the Sprint network “failed” once per week during maintenance hours. Those failures are probably due to planned software upgrades of all routers in the network.

When an IS-IS[1] router needs to stop forwarding IP packets, IS-IS can flood a new LSP indicating the router as overloaded [ISO02]. Some ISPs have even defined operational procedures [DFM04] to bring routers down by changing link metrics and setting the `overload bit`, but those procedures are not sufficient to ensure that transient loops will not occur during the IGP convergence. The graceful restart extensions [SDV02, SG04b, MPEL03] could be used when a router is rebooting. However, those extensions cannot be used for the maintenance operations affecting the forwarding plane of the router. As shown by the above discussion, there are many different types of changes in IP networks that should be handled without risking to create transient routing loops in the network.

Another kind of events to which routers should adapt without packet loss are the "positive" events. When a router or a link is brought up in the topology, it is very unfortunate that the IGP converges by letting packets be dropped.

4.5 Conclusion

In this chapter, we presented a method to count the potential forwarding loops that can occur upon a single link failure, given a network topology, i.e. the network graph and the metrics associated with the links of the network. Then, we applied our technique to four different topologies. We observed that micro loops can actually occur in real ISP topologies, due to the way these are designed. We also notice that pathological cases can occur, where a large amount of destinations are affected by microloops upon a single link failure. This observation is somehow concerning as Quality of Service can be deeply impacted as a result of these micro loops.

When a link is manually shut down by an operator, the same micro loops can occur so that manual, predictable operations on the network can also have an harmful impact on the reachability throughout the network. This issue strongly motivates the introduction of loop avoidance mechanism that can be used upon manual topology reconfiguration, where there should not be a "recovery" from failure, but,

[1]A similar reasoning is valid for OSPF as well.

instead, a smooth transition to a forwarding state that does not use the link being manually removed.

Also, in the case of a sudden failure protected by a Fast Reroute mechanism, the reachability in the network is ensured, so that the transition to the forwarding state taking into account the new shortest paths accross the network should also be carried out in a smooth, loss free fashion.

Chapter 5

Forwarding Loop avoidance using Ordered FIB Updates

When using link-state protocols such as OSPF or IS-IS, forwarding loops can occur transiently when the routers adapt their forwarding tables as a response to a topological change. In chapter 4, we analyzed the potentiality of such forwarding loops on real ISP topologies, and we motivated loop avoidance mechanisms for the IGP.

In this chapter, we present a mechanism that lets the network converge to its optimal forwarding state without risking any transient loops and the related packet loss. The mechanism is based on an ordering of the updates of the forwarding tables of the routers. Our solution can be used in the case of a planned change in the state of a set of links and in the case of unpredictable changes when combined with a local protection scheme.

The supported topology changes are link transitions from up to down, down to up, and updates of link metrics. Finally, we show by simulation that sub-second loop free convergence is possible.

5.1 Introduction

In this chapter, we first prove in section 5.2 that the updates of the FIB can be ordered to avoid transient loops after a topology change affecting a set of links. This proof is constructive as we give an algorithm that routers can apply to compute the ranks that let them respect the proposed ordering. To respect this ordering, routers can compute a “rank” corresponding to the time at which they must update their FIB. In section 5.3, we analyse the ranks that routers would apply in real topologies upon a link shutdown. Next, in section 5.4, we propose to use "completion messages" to bypass the ranks computed by the routers, so that the loopfree convergence process can complete faster. In section 5.5, we evaluate by simulations the time required by our modified link-state protocol to converge. In Section 5.6, we present an optimization that lets routers find out when they can reroute without respecting their rank while ensuring that no loop will occur. In section 5.8, we summarize the applicability of the solution w.r.t. the type of routing system being used and the nature of the topological change. We notably discuss the applicability of the solution when the topological change is a sudden failure of resources that are not protected by a Fast Reroute technique. In section 5.7, we review the other mechanisms that have been proposed to enhance the convergence of the IGP. Finally, we draw some conclusions in section 5.9.

5.2 An ordering for the FIB updates

To avoid transient loops during the convergence of link-state protocols, we propose to force the routers to update their FIB by respecting an ordering that will ensure the consistency among the FIB of all the routers during the whole convergence phase of the network.

In the context of a predictable maintenance operation, the ressources undergoing the maintenance will be kept up until the routers have updated their FIB and no longer use the links to forward packets. In the case of a sudden failure of a link that is protected with a Fast Reroute technique, the proposed ordering ensures that a packet entering the network will either follow a consistent path to its destination by avoiding the failed component or reach the router adjacent to the failure and will be deviated by the Fast Reroute technique to a node that is not affected by the failure, so that it will finally reach its destination.

In this section, we briefly review the orderings in the case of single link events (link down or metric increase, link up or metric decrease), that we proposed in [FB05]. Then, we extend the solution to events affecting Shared Risk Link Groups. Finally, we discuss router and line card events, which are particular SRLG cases.

As those orderings are applied in the case of **predictable changes** and in the case of sudden changes where a local protection is provided, avoiding transient loops will permit to avoid all the packet losses during the IGP convergence inside the network.

Note that the proposed orderings are valid when asymmetrical link metrics are used in the topology, i.e., when there exists links $X \leftrightarrow Y$ such that the metric of $X \rightarrow Y$ is not equal to the metric of $Y \rightarrow X$.

Also, the solution takes into account the case where multiple equal cost paths from one router to another are used before and/or after the event. In the following sections, we use the terms of Shortest Path Trees, and reverse Shortest Path Trees to respectivley denote the set of shortest paths from a router to the other routers of the network and the set of shortest paths from all the routers to a given router. When Equal Cost MultiPath (ECMP) is used, the union of these paths form an acyclic graph, not a tree. We will explain how routers deal with this when it could lead to ambiguous results in the provided proofs and algorithms.

5.2.1 Single Link Events

Link down or metric increase

In the case of a link down or metric increase event for a link $X \rightarrow Y$, a router R must update its FIB **after all the routers that used** R to reach Y **before** the event.

To respect this ordering, R computes $rSPT_{old}(X \rightarrow Y)$, the part of the reverse Shortest Path Tree (rSPT) of Y in the old topology that is affected by the change. The rSPT of a node is the set of shortest paths to this node. The part of interest in this rSPT is the set of shortest paths *to* Y that are affected by the failure of $X \rightarrow Y$. Within this part, the subtree that is under R in $rSPT_{old}(X \rightarrow Y)$ contains all the paths to R that were used to reach at least one destination via R and link $X \rightarrow Y$ before the event.

The rank of R is equal to the depth of this subtree, $depth(R, rSPT_{old}(X \rightarrow Y))$. In the case of ECMP, the rank of R is the maximum number of hops among the equal cost shortest paths to R inside the graph. This value can be easily obtained by computing $rSPT_{old}(Y)$, the set of shortest paths to Y.

The time at which R will be allowed to update its FIB is equal to the obtained rank multiplied by a configurable worst-case FIB update time, that depends on the number of prefixes that are advertised in the network.

By applying this ordering, a router R that has not yet updated its FIB for the destinations that it reached via $X \rightarrow Y$ will forward packets to these destinations along routers that computed a larger rank value, so that the ordering will be respected.

1. Let us assume that a router R was using a neighbor N to reach Y via X
2. R is below N in $rSPT_{old}(X \rightarrow Y)$
3. From 2, we have

$$Rank(N) = depth(N, rSPT_{old}(X \rightarrow Y))$$

$$\geq$$

$$depth(R, rSPT_{old}(X \rightarrow Y)) + 1$$

4. The same property can be verified hop by hop along the paths from R to X

We proved that the proposed rank will let a router R update its FIB before the routers that R used to reach the failing link. This implies that the routers along those paths will not have updated their FIB when R has not updated its FIB yet. The packets forwarded by R will thus arrive in Y and be forwarded on non affected paths from Y to d. It is sure that the paths from Y to d are not affected by the event. Indeed, if one router was using $X \rightarrow Y$ to reach d, then Y could not use $X \leftrightarrow Y$

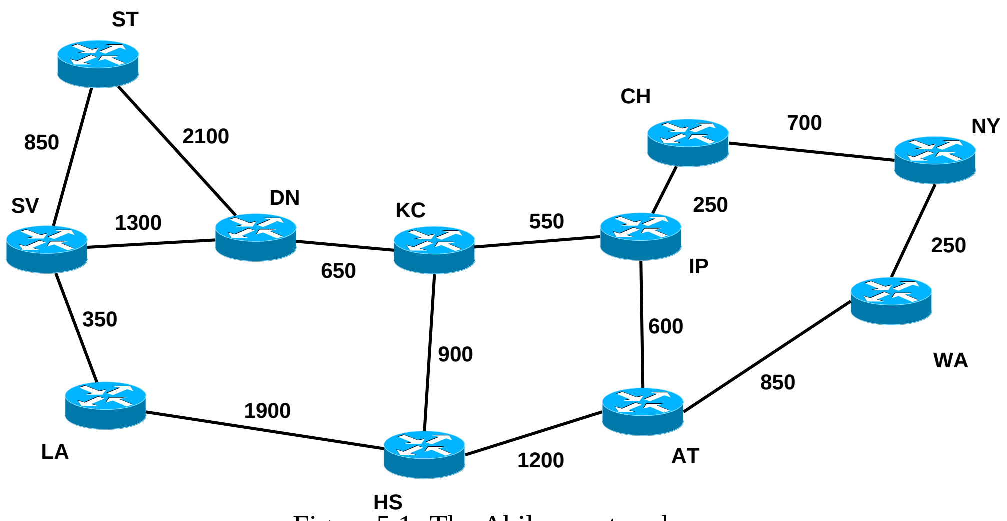


Figure 5.1: The Abilene network

to reach d. The contrary would imply an intra domain forwarding loop while the network was stable.

As an example, let us consider the shutdown of link $IP \leftrightarrow KC$ in figure 5.1. According to the ordering, the rank of IP is 3, as longest branch under IP in $rSPT_{old}(IP \rightarrow KC)$ is $IP - CH - NY - WA$. AT has a rank of 0, because it is a leaf in $rSPT_{old}(IP \rightarrow KC)$. So, IP will reroute after AT and no loop will occur along $IP \leftrightarrow AT$. Similarly, the rank of NY is one because the deepest branch under NY in $rSPT_{old}(IP \rightarrow KC)$ is $NY - WA$. WA has a rank of 0, as it is a leaf in $rSPT_{old}(IP \rightarrow KC)$. So, WA will update its FIB before NY and no loop will occur along $WA \leftrightarrow NY$.

Link up or metric decrease

When a link $X \rightarrow Y$ is brought up in the network, or its metric is decreased, the required ordering is such that a router R updates its FIB **before the routers that will use** R to reach Y via X. To apply this ordering, R computes $PathLength(R, X)$, the number of hops of its path from R to X. Note that in the case of ECMP, the considered number of hops is the largest one among the multiple equal cost paths. This value, that we call the rank of R, is easily obtained by R when it computes its new SPT to update its FIB.

All the routers N along the paths from R to X compute a shorter rank value, so that they will update their FIB before R, and the ordering will be respected.

For each router N on the path from R to X :

1.
$$Rank(R) = PathLength(R, X)$$
$$\geq$$

$$PathLength(R, N) + PathLength(N, X)$$

2.
$$Rank(N) = PathLength(N, X)$$

3. $PathLength(R, N) > 0$

4. From 2 and 3, we have

$$PathLength(R, X) = Rank(R)$$
$$>$$
$$PathLength(N, X) = Rank(N)$$

The time at which R will be allowed to update its FIB is equal to its rank multiplied by the worst-case FIB update time.

We proved that each router N being on the new paths from R to X will update its FIB before R. Thus, packets rerouted by R towards $X \rightarrow Y$ will be forwarded by routers with updated FIBs, so that the packets deviated by R will reach $X \rightarrow Y$ to finally reach their destination.

As an example, let us consider the re-activation of link $KC \leftrightarrow IP$ in the topology depicted in figure 5.1. There could be a forwarding loop in that case if WA updates its FIB with regard to this event before NY, as WA would forward packets destined to KC along $WA \rightarrow NY$, although NY was forwarding such packets along $NY \rightarrow WA$ before the link up event. Also, a forwarding loop could take place along $AT \leftrightarrow IP$ if AT updates its FIB before IP. However, this second forwarding loop should not happen in practice because IP will be the first to be aware of the link up event. According to the proposed ranking, IP updates its FIB directly because $PathLength(IP, IP) = 0$. AT, will update its FIB after one worst-case FIB update time, as $PathLength(AT, IP) = 1$. Similarly, WA will update its FIB after NY because $PathLength(NY, IP) = 2$ and $PathLength(WA, IP) = 3$, so that the potential loop between NY and WA could not occur if the ranking is applied.

5.2.2 Shared Risk Link Group events

In this section, we extend the idea underlying the scheme for single link cases to predictable events affecting a set of links in the network.

One could argue that when an operator wants to shut a set of links down, he could consecutively shut down each link of the set and let IS-IS apply the solution for single link events.

This technique has some disadvantages. Firstly, this technique can produce a large number of end-to-end paths shifts, as routers may, as a response to the shutdown of a link, reroute packets on alternate paths via other links to be shut down.

The techniques proposed in this section let routers use their post-convergence outgoing interfaces towards a given destination upon the first and unique update of their FIB for this destination. Secondly, predictable events affecting multiple links can be caused for example by the installation or the shutdown of an optical switch supporting a set of links in the network. As the optical layer and the IP network tends to be more and more integrated, an optical switch undergoing a shutdown could notify the IS-IS routers to which it is connected of its upcoming failure. In this case, the event is not under the control of the operator of the IP network so that it would not be possible for the operator to schedule a sequence of single link shut down operations.

These two issues motivated the generalization of our techniques to the events affecting a set of links.

Currently, IS-IS does not allow to perform a shutdown or installation of a set of links, using a single command issued in one router, or by flooding one single routing message. Indeed, to describe the failure of an SRLG, it is required that at least one router adjacent to each of the links of the SRLG floods a link-state packet describing the failure of this link. The only cases where this is possible is for the particular SRLG cases being the set of links connected to one router. But this does not cover the case of a shutdown or installation of an optical switch connected to a set of routers. We thus need to introduce the possibility to send IS-IS or OSPF messages stating that a given SRLG is going to be shut down or brought up in the network as a result of the event occuring at the optical level. This could be achieved by assigning SRLG IDs to the links of the network and let each router describe the "shared state" of the SRLG to which its links belong. In order to consider a given SRLG as being up, all the advertised shared states associated with this SRLG must be set to up by the routers that are adjacent to one member of this SRLG. To manually shut down a set of links, an operator could then issue a command in one router adjacent to the members of the SRLG, so that the router will flood its Link-State Packet by setting the state of this SRLG to down.

Note that we do not cover the case where a set of unrelated sudden link failures occur concurrently in the network. When routers face this situation they should, as described in [FBS$^+$06], fall back to the regular, fast convergence process.

In the remainder of this section we describe how routers can adapt to the manual shut down of a set of links by avoiding transient loops. Next, we present the solution when a set of links comes back up in the network. Finally, we consider the operational case of an SRLG whose links are connected to one common node. These specific cases cover router shut down and installation, as well as line card shutdown and installation.

SRLG Shutdown or SRLG metric increase

In this section, we propose an ordering of the FIB updates that preserves the transient forwarding consistency among the routers of the network, in the case of a metric increase (or shutdown) of a set of links. We firstly give a property of the transient forwarding states that allows a loop-free convergence, and then we present an ordering that permits to respect this property. As we present the solution in the context of a predictable topology change, we can assume that the links affected by the shut down operation remain up until the routers adjacent to these no longer forward packets along those links, i.e., the routers will keep the link up until they have updated their FIB.

The idea underlying the scheme is the same as for the single link case. We want to ensure that, during the whole convergence phase, if a packet with destination d arrives at a rerouting router R that has not yet updated its FIB for d, then all the routers along the paths from R to d have not yet updated their FIB for d either. This implies that *once a packet reaches a rerouting router with an outdated FIB for its destination, it will follow an outdated but consistent path towards it.*

If this property is always verified, no transient loop can occur, as each packet entering the network will first follow a path that contains a sequence of routers with an updated FIB. Then, either it reaches its destination or it reaches a router with an outdated FIB. In the later case, we know from the preceding paragraph that the packet will reach its destination. Thus, we know that each packet entering the network follows a loop-free path towards its destination if the proposed ordering is respected.

To ensure the respect of this ordering using a rank, the ranking must be such that if a router R updates its FIB for a destination d with a rank r, then all the routers lying on the initial paths from R to d that must update their FIB for destination d, must do so with a rank that is strictly greater than r. We propose such a rank in Definition 5.2.1.

Definition 5.2.1 *The rank function for the shut down of a set of links* $\{l_1, l_2, ..., l_j\}$*, is* $min\{depth(R, rSPT_{old}(l_k) \mid l_k \in Paths(R, d)\}$*, with* $Paths(R, d)$ *being the set of paths that are used by* R *to reach* d *before the event.*

In other words, a router computes the rank associated with each individual link being shut down that it is currently using, as defined for the single link shutdown problem. For each destination for which it has to perform a FIB update, it applies a rank being the minimum among the ranks associated with the links that it uses to reach this particular destination.

$rSPT_{old}(l_k)$ is the acyclic graph containing all the shortest paths towards the tail-end of link l_k on the topology before the event. $depth(R, rSPT_{old}(l_k))$, is the maximum hop distance among the paths to R in this acyclic graph. This depth can be easily computed on the fly of a reverse SPT computation with the tail-end of l_k as a root.

Theorem 5.2.2 *The rank proposed in Definition 5.2.1 satisfies the required ordering of the FIB updates.*

Let us now prove Theorem 5.2.2.

Proof : Let us consider that a router R updates its FIB for a destination d with a rank ($Rank(R, d)$).

We have to prove that, for a router N lying on the initial paths from R to d we have $Rank(R, d) < Rank(N, d)$.

Let us denote the affected links on the paths between N and d by $\{l_1, l_2, ..., l_s\}$.

According to the definition of an SPT, we can see that all the affected links on the paths between N and d are also on the paths between R and d, as R has N on its shortest paths towards d. Note that R can also have other affected links on its paths towards d. These are the affected links used by R to reach N, and the affected links that are on other equal cost paths to d than the ones via N. We denote the links that are used by R and not by N to reach d by $\{l_{s+1}, l_{s+2}, ..., l_{s+t}\}$.

1. From the definition of a rank we have

$$Rank(N, d) = \min_{1 \leq i \leq s}(depth(N, rSPT(l_i))),$$

and

$$Rank(R, d) = \min_{1 \leq i \leq s+t}(depth(R, rSPT(l_i))).$$

2. As, before the event, R uses N to reach d, and N uses $l_{1...s}$ to reach d, we have that R uses N to reach $l_{1...s}$, so that R is below N in $rSPT_{old}(l_i),\ with\, 1 \leq i \leq s$, and thus

$$\forall i : 1 \leq i \leq s :$$
$$depth(R, rSPT_{old}(l_i)) < depth(N, rSPT_{old}(l_i))$$

So that we have $Rank(R, d) < Rank(N, d)$. ∎

Thus, *the rank to reroute for destination d in a router R, according to the failure (or the metric increase) of a set of links $l_1, l_2, ..., l_s$ is $min\{depth(R, rSPT(l_v)) \mid l_v \in Paths(R, d)\}$.*

Note that each destination is associated with a rank whose value belongs to the set of ranks computed for each failing link, so that in the worst-case, the FIB updates will be split in as many parts as there are links being shut down.

Let us illustrate with figure 5.2 the various properties that lead to a loop free convergence when the proposed ranking is respected. In this figure, the links $R \leftrightarrow Y$, $Y \leftrightarrow Z$, $S \leftrightarrow T$, and $T \leftrightarrow Z$ are being shut down. Initially, R is using N to reach destination d, so that to apply the ordering, R should have a rank strictly lower than the rank of N w.r.t. destination d.

All the affected links that N uses to reach d, i.e., $S \rightarrow T$ and $T \rightarrow Z$, are used by R to reach d, because R uses N to reach d. R also has other affected links in its paths towards, d; $R \rightarrow Y$ and $Y \rightarrow Z$. N will consider its rank as being the minimum between the depths of the two branches under N in $rSPT(S \rightarrow T)$

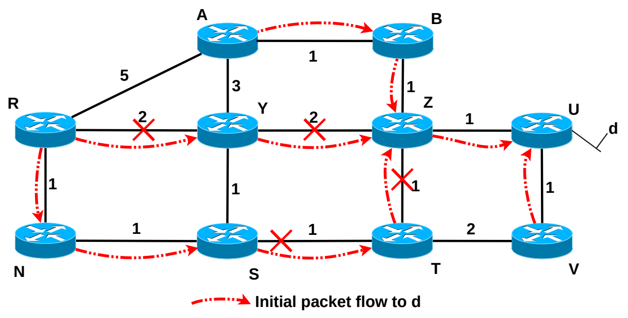


Figure 5.2: Illustration of the SRLG down case

and $rSPT(T \rightarrow Z)$. R will consider its rank between the depths of the four branches under R in $rSPT(S \rightarrow T)$, $rSPT(T \rightarrow Z)$, $rSPT(R \rightarrow Y)$ and $rSPT(Y \rightarrow Z)$. R is below N in $rSPT(S \rightarrow T)$, so that the rank associated by R to this link is strictly lower than the one associated by N to the same link. The same reasonning can be applied for link $T \rightarrow Z$. So, R could not have a rank larger or equal to the rank of N w.r.t. destination d, as R will use as its rank the minimum depth among those of the branches under itself in these two rSPTs and also in the branches below R in $rSPT(R \rightarrow Y)$ and $rSPT(Y \rightarrow Z)$.

SRLG up event or metric decrease

When a set of links is brought up in the network, or when the metrics of a set of links are decreased, routers can also apply a rerouting scheme that ensures the transient forwarding consistency during the whole convergence phase that follows the event.

The proposed scheme allows a rerouting router R to update its FIB for a destination d once all the routers along the paths from R to d have updated their FIB for d.

If this property is always verified, no transient loop can occur, as each forwarded packet for a given destination d will first follow a path composed of a set of routers whose FIBs have not been updated yet for d. Then, either it reaches d, or it reaches a router R that has already updated its FIB for d. In the later case, we know that all the routers on the path from R to d have updated their FIB for d, so that the packet will be consistently forwarded to d.

Now, we show how routers can apply the proposed ordering.

In the case of a single link $X \rightarrow Y$ being brought up, a rerouting router R updates its FIB by respecting a rank equal to the length (in hops) of its new shortest path to X.

In the multiple link case, a router can have a new SPT such that the shortest paths towards a destination d can contain several of the affected links. However, R will still compute the ranks associated with each link being brought up individually.

Then, for each destination d, it will apply a rank equal to the maximum of the ranks among those associated with the affected links that it will use to reach d.

Let us prove that this technique verifies the aimed loop-free property.

Let us consider that a router R updates its FIB for a destination d. We have to prove that for a neighbor N of R lying on the new paths from R to d, we have $Rank(R, d) > Rank(N, d)$.

According to the definition of a SPT, we can see that all the links of the considered SRLG that are on the new paths from N to d are also on the new paths from R to d, as R will use N to reach d. We will denote those links by $\{l_1, l_2, \dots, l_s\}$. R can also have other links of this SRLG in its new paths towards d. It could be, for example, $R \rightarrow N$, or links on another equal cost path towards d. We will denote them by $\{l_{s+1}, l_{s+2}, \dots, l_{s+t}\}$.

As R will use N to reach d, and N will use $l_{1\dots s}$ to reach d, we have that R will use N to reach $l_{1\dots s}$, so that the rank that R associates with l_i is at least equal to $PathLength(R, N) + PathLength(N, head\_end(l_i))$, i.e., the maximum hop length among the shortest paths from R to N plus the rank that N associates with l_i, which is the maximum hop length among the shortest paths from N to the head end of the link l_i, i.e, X if $l_i = X \rightarrow Y$. This gives the maximum hop length among the shortest paths (considering the IGP metrics) from R to the head end of l_i via N.

From the following properties,

1. $Rank(N, d) = \max_{1 \le i \le s}(PathLength(N, head\_end(l_i)))$

2. $Rank(R, d) = \max_{1 \le i \le s+t}(PathLength(R, head\_end(l_i)))$,

3. $\forall i : 1 \le i \le s :$

$$PathLength(R, head\_end(l_i))$$

$$>$$

$$PathLength(N, head\_end(l_i))$$

So that we have $Rank(R, d) > Rank(N, d)$

The same property can be recursively discovered between N and its nexthops towards d, so that we prove that the rank applied by R for d will be greater then the rank applied by each router on new paths from R to d.

As the rank that a router applies for a destination d belongs to the set of ranks that the router computed for each affected link, the number of distinct ranks that can be applied by a router is bounded by the number affected links.

5.2.3 Router and Linecard events

Among the events concerning sets of links, we can find particular predictable events for sets of links connected to a single router. This is the case for router shut down and setup events, and for line card removal or installation. These kind of events are easy to identify as a set in IS-IS if, upon the shutdown of the router, the IS-IS overload bit is set and a link-state packet is flooded by the concerned router. In the case of a router or line card up, the event can be easily identified as a set if the router sends a link-state packet describing all the links being enabled.

In such specific SRLG cases, the first possible behavior of the routers is to consider the event as any other set of link events, and apply the mechanism proposed for the general SRLG cases. However, a simpler behavior is applicable, which will let each router compute one single rank and perform its FIB update in one shot.

When a router or a line card of X is shut down, the behavior is similar to a link down event. The rank computed by a router R is equal to the depth of the tree below R in $rSPT_{old}(X)$.

When a router X or a line card of X is brought up in the network, the behavior is similar to a link up event. The rank computed by a router R is equal to the maximum length (in hops) of the new paths from R to X. The proofs are very similar to the ones provided for the single link events. We omit them for the sake of brievety.

5.3 Analysis of the rank based ordering in ISP topologies

If the ordering of the FIB updates is ensured by the means of a timer whose value is set according to a rank and a worst-case FIB update time, the delaying of the FIB updates can be long if the topology is such that large rank values could be computed by the routers for some events.

To analyze this, we computed the ranks that routers would apply in the case of single link failures. For each link shutdown, we looked at the rank applied by the router being the head-end of the link being shutdown. This router is the one with the largest rank for the considered event. The rank that is applied by this router is equal to the worst-case rank that would be applied when the link is brought back up in the topology, so that the figure for the link up cases is the same.

In Figure 5.3, we present the ranks associated with the links of a Tier-1 ISP, containing about 800 (directed) links and about 200 nodes. Note that among those links, the IGP metrics are such that some links are not used and a few others are used only in one direction. The ranks associated with those unused links are equal to 0 in the figure. Note that some links have a rank of 0 even if they are used. This is typically the case of a link from an access router to a core router that is only used by the access router itself. From this figure, we can see that some paths are 14 hops long. Moreover, a large number of prefixes are advertised in this network, so that the worst case FIB update time could be set quite long in order to be conservative. If the worst case FIB update time were set to 1 second, the maintenance of a link

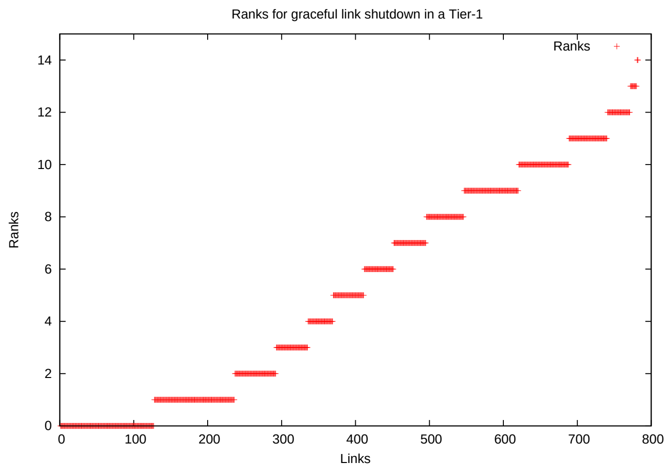


Figure 5.3: Ranks for the shutdown of the links in a Tier-1 ISP

in this network could last up to 14 seconds. This could be considered too long by operators, as other events could occur within such a time window.

However, in the case of a maintenance of a link terminating those 14 hops paths, very few routers using the link are rerouting routers. This means that the FIB update time allocated to them is a waste of time, as routers will not perform FIB updates during those periods. The effect is the same in the case of a link up event.

We performed the same analysis on Geant, a network containing 72 (directed) links and 22 nodes [GEA]. We learned from this analysis that 20 of the 72 directed links were only used by the head-end of the link, so that the obtained rank was 0. No delaying would be applied if those links were shut down, and the link could be effectively shut down just after the FIB update performed by the head-end of the link. The worst-case rank is 4, and was obtained for 7 links. So, even with a very conservative worst-case FIB update time of 1 second and no completion messages, the maintenance of a link in Geant would cause a transiently loop free convergence time of 4 seconds.

This long convergence time motivated the introduction of completion messages to shortcut the delaying allocated to the routers as soon as possible [FB05].

5.4 Completion Messages to speed up the convergence phase

One issue of the rank based ordering scheme is that it assumes a worst-case FIB update time in each router taking part in the process. However, in many cases, routers only have to perform a FIB update for a subset of the reachable destinations, if any. Moreover, the performances of the routers in a network can differ, so that the assumed worst-case FIB update time could be artificially long. In summary, the timer-based ordering works, but it tends to unnecessarily delay the FIB updates in the routers.

To solve that issue, we introduce completion messages [FB05]. These messages can be placed inside IS-IS Hello PDUs [BFSP06]. They are sent by routers to their neighbors to announce that they have performed their FIB update by respecting the ordering. When computing its rank, a router implicitly computes the set of neighbors from which a completion message should be received before it can update its own FIB. Routers will retain this set in a "Waiting List".

In this section, we explain how such lists can be built, and when routers are allowed to send completion messages to their neighbors, by still ensuring the proposed loop free ordering of the FIBs.

We firstly present the scheme for single link events, and then we generalize the solution to events affecting sets of links.

5.4.1 Single Link Events

Link down or metric increase

In the case of a link $X \rightarrow Y$ down or metric increase event, a router R computes $rSPT_{old}(X \rightarrow Y)$ to obtain its rank. By doing this, it also computes the set of its neighbors that were using it to reach Y. This set of neighbors will compose the waiting list of R. When this waiting list empty, i.e., when $Rank(R) = 0$, R can update its FIB directly. When a router has updated its FIB, it sends a completion message to the neighbors that it was using to reach $X \rightarrow Y$. When a router R receives a completion message from one neighbor, it removes the sender from its waiting list. When the waiting list of R becomes empty, it is allowed to update its FIB and send its own completion message.

When a router receives a completion message from a neighbor, it knows that the sender has updated its FIB by respecting the ordering. Indeed, the sender could only send the completion message because the computed delay for its FIB update obtained by the ranking has elapsed or because its Waiting List has been emptied. In other words, when the Waiting List of a router R becomes empty, all the routers that were using R to reach $X \rightarrow Y$ have sent their completion message, so that all of them have updated their FIB.

Link up or metric decrease

In the case of a link $X \rightarrow Y$ up or metric decrease event, a router R recomputes $SPT(R)$ to determine the FIB updates that are required and its rank. If $X \rightarrow Y$ is in its new SPT, R will have to reroute after its nexthops for X. Those nexthops will compose its waiting list for the event. When a router updates its FIB, it will send a completion message to its neighbors. When a router receives a completion message from one neighbor, it removes the sender from its waiting list. When the Waiting List becomes empty, it is allowed to update its FIB and send its own completion message.

The ordering is still respected as if the Waiting List of a router R is empty, all the routers on the paths from R to $X \rightarrow Y$ have sent their completion message, so that all of them have updated their FIB.

5.4.2 Shared Risk Link Group events

SRLG down or SRLG metric increase

Each router will maintain one waiting list associated with each link being shut down during the rSPT computations. A rerouting router R will update its FIB for a destination d (which means that its paths to d contain one or more links of the SRLG) once it has received the completion messages that unlock the FIB update in R for one of the links being shut down. When updating its FIB, R selects the outgoing interfaces for destination d according to the new topology, i.e., by considering the removal or the metric increase of all the affected links.

The meaning of a completion message concerning a link l sent by a router R is that R has updated its FIB for all the destinations that it was reaching via l before the event.

Let us now show that if a packet with destination d reaches a rerouting router R that has not performed its FIB update for destination d, then all the routers on its paths to d cannot have performed a FIB update for d.

If R has not updated its FIB for destination d, it cannot have sent a completion message for any of the failing links l that it uses to reach d. The failing links that a router N on $Paths_{old}(R, d)$ uses to reach d are used by R to reach d, so that N cannot have received all the necessary completion messages for any of those links. In other words, R did not send a completion message for the links that it uses to reach d. Thus R locks the FIB update for those links along its paths towards them.

In Figure 2, we provide the pseudocode that implements the ordering with completion messages. To process the metric increase (or shutdown) of a set of link S, a router R will compute the reverse SPT rooted on each link l belonging to S, that it uses in its current, outdated SPT. During this computation, it will obtain the rank associated with l. It will then record the nexthops that it uses to reach l in a list $I(l)$. These are the neighbors to which it will send a completion message concerning link l. If the rank associated with a link is equal to zero, then R updates its FIB directly for the destinations that it reaches via this link, and it sends a completion message to the corresponding nexthops. In the other cases, R builds the waiting list associated with l, containing the neighbors that are using R to reach l, and it starts the timer considering the rank associated with this link.

Once a waiting list for a link l becomes empty or its associated timer elapses, R can update its FIB for all the destinations that it reached via this link and send its own completion message $CM(l)$ towards the neighbors that it used to reach the link.

```
Metric increase event for a set of Link $S$ processed by router R:
//Computation of the rSPTs of the affected links used by R
foreach Link $X \rightarrow Y \in S$ do
    if $X \rightarrow Y \in SPT_{old}(R)$ then
        //Computation of the rSPT
        LinkRSPT = rSPT($X \rightarrow Y$);
        //Computation of the rank
        LinkRank = $depth(R, LinkRSPT)$;
        //Computation of the set of neighbors to which a
        //completion message concerning this link will be sent
        $I(X \rightarrow Y)$=$Nexthops(R, X \rightarrow Y)$;
        if LinkRank ==0 then
            //R is a leaf in rSPT($X \rightarrow Y$),
            //it can update its FIB directly
            foreach d : $X \rightarrow Y \in Path_{old}(R, d)$ do
                UpdateFIB(d);
            end
            //R can send its completion message for this link.
            foreach $N \in I(X \rightarrow Y)$ do
                send($N, CM(X \rightarrow Y)$);
            end
        end
        else
            //R is not a leaf in rSPT($X \rightarrow Y$),
            //Computation of the waiting list. WaitingList($X \rightarrow Y$)=
            Childs(R,LinkRSPT);
            //Start the timer associated with this link.
            StartTimer($X \rightarrow Y$, LinkRank * MAXFIBTIME);
        end
    end
end

Upon reception of $CM(X \rightarrow Y)$ from Neighbor $N$ :
WaitingList($X \rightarrow Y$).remove($N$);

Upon (WaitingList($X \rightarrow Y$).becomesEmpty() ||
   Timer($X \rightarrow Y$).hasExpired()) :
//All the necessary completion messages have been received for
//the link or the timer associated with this link has expired
//Update the FIB for each destination that was reached
//via this link.
foreach d : $X \rightarrow Y \in Path(R, d)$ do
    UpdateFIB(d);
end
//Send the completion messages to the neighbors that were
//used to reach this link.
foreach $N \in I(X \rightarrow Y)$ do
    send($N, CM(X \rightarrow Y)$);
end
```

Algorithm 2: Processing of a set of link metric increase events

SRLG up or SRLG metric decrease

In the case of a set of link up or link metric decrease events, each router will maintain a Waiting List associated with each link being brought up in the network. For each affected link, its associated Waiting List is the same as for the single link case.

A router R is allowed to reroute packets for a destination d to a new nexthop N when it has received the completion messages from N associated with all the affected links of at least one of the equal cost paths between N and d in the new SPT of R.

A router R will send completion messages for a link $X \rightarrow Y$ to its neighbors once it has updated its FIB for the destinations that it reaches via $X \rightarrow Y$ and the affected links for which it already sent a completion message. Note that if there are some destinations that R now reaches via $X \rightarrow Y$ and some other upcoming links, the fact that R sent a completion message for the link $X \rightarrow Y$ does not mean that R has updated its FIB for this destination. It means that R has updated its FIB for the destinations that are only reached via the new upcoming link $X \rightarrow Y$. When a router has sent completion messages for a set of upcoming links S, it means that it has updated its FIB for all the destinations that it reaches via any subset of S.

When there are equal cost paths between N and d, R has the choice to deviate packets destined to d towards N when N has sent the completion messages associated with all the upcoming links on all those paths, or when N has sent the completion messages associated with all the upcoming links belonging to at least one of those equal cost paths.

In Figure 3, we present the pseudocode that implements the ordering with completion messages. We only present the one which allows a FIB update for a destination d in a router R, towards a new neighbor N, as soon as N uses one of its post-convergence equal cost paths towards d.

To process the metric decrease (or the installation) of a set of links S, a router R will compute SPT_{new} to obtain the FIB updates that must be performed. Then, the router initializes a set ($Rerouted$) containing the destinations for which an update has already been sent to the line cards, and a set ($CMSent$), containing the set of upcoming links for which it has already sent a completion message. The first set is useful if more than one new outgoing interfaces will be used for some destinations. The second set will permit to avoid sending duplicates of completion messages.

R must then build the waiting lists associated with each of the affected links that it will use. When R receives a completion message for a link $X \rightarrow Y$, it applies the procedure $followNewSPT$. This procedure will perform the FIB updates that are unlocked by the reception of the completion message. The reception of $CM(X \rightarrow Y)$ from N means that N is using at least one post-convergence path for the destinations that are below $X \rightarrow Y$ in $SPT(N)$. It also means that N does not use any outdated path towards those destinations. R can thus follow its own SPT and deviate to N the packets towards the destinations that it will reach via N and $X \rightarrow Y$. The SPT will be followed from $X \rightarrow Y$ until R reaches another upcoming link within this part of its SPT. At that time, if a completion message concerning this link had already been received from N, then R is allowed to follow its SPT further on and perform the unlocked FIB updates.

The first time a new nexthop for a destination d is installed in the FIB of a router, all the nexthops that will no longer be used to reach d are removed from its FIB. If an additional (equal cost) nexthop is discovered later for d, it will simply be added because d will belong to $Rerouted$ at that time.

The first time an upcoming link is followed by the $followNewSPT$ procedure, and the corresponding updates are performed, the router will send a completion message for this link. If the link is followed again, because the router has multiple paths towards this link, no additional completion message will be sent because the link will belong to $CMSent$ at that time.

5.4.3 Router and Line card events

Router and Line card down events

Let us consider that a line card of a router X is to be removed, or that X is to be shut down.

The waiting list of a router R for such an event contains the neighbors of R that are below R in $rSPT_{old}(X)$. These are the neighbors of R that were using R to reach X. If R is a leaf in $rSPT_{old}(X)$, it is allowed to update its FIB directly, and send a completion message to its nexthops for X. If R is not a leaf, then it waits for completion messages from its neighbors. When a router R receives a completion message specifying the router or line card down event in X, it removes the sender from its Waiting List. When this Waiting List becomes empty, R is allowed to perform its FIB update and then send its own completion messages to its nexthops to X.

When X has received the completion messages from all its neighbors, it is allowed to actually shut itself down or shut the line card down. During the whole convergence phase, when a packet reaches a router R that has not updated its FIB for this destination, its nexthops for this destination did not receive a completion message from R, so that they also have outdated FIB. This property can be verified hop by hop along the path from R to X, so that the packet will reach X and be forwarded to a neighbor of X whose paths towards the destination is not affected by the event.

Router and Line card up events

When a router X or a line card of X is brought up in the network, the Waiting List of a router R contains the neighbors of R that R will use to reach X. X will be the first router to update its FIB, and will send a completion message to all its neighbors. When a router R receives a completion messages specifying the router or line card up event in X, it removes the sender from its Waiting List. When this

```
Metric decrease event for a set of Link $S$ processed by router R:
$SPT_{new}$ = recomputeSPT();
//Compute the set of updates that will be performed on the FIB
nexthopsUpdates = getNexthopUpdates($SPT_{new}$);
//Initialize the set of Link in $S$ for which a completion message has been sent.
CMSent = {};
foreach Link $X \rightarrow Y \in S\ : X \rightarrow Y \in SPT_{new}$ do
    //Get the nexthops used to reach the upcoming links.
    //The new nexthops are used if these have changed
    WaitingList($X \rightarrow Y$) = getNexthops($X$);
end

Upon reception of CM($X \rightarrow Y$) from neighbor $N$ :
WaitingList($X \rightarrow Y$).remove($N$);
//Perform the updates that are unlocked by this completion message;
if $X \rightarrow Y \in SPT_{new}$ and $X$ reached via $N$ then
    followNewSPT(Y,N);
    if not CMSent.contains($X \rightarrow Y$) then
        SendToNeighbors(CM($X \rightarrow Y$));
        CMSent.add($X \rightarrow Y$);
    end
end

followNewSPT(Y,N):
//Explore the graph and perform the necessary FIB updates
if nexthopUpdates.contains(destination Y, nexthop N) then
    //Add nexthop N for destination Y.
    //First call to SendFIBUpdateToLC(Y, .) will remove
    //the nexthops that are no longer used
    //to reach Y from the FIB in the LineCards
    SendFIBUpdateToLC(Y,N);
end
//FIB updated for destination Y if needed,
//Update the FIB for the destinations behind Y in the new SPT.
foreach Link $Y \rightarrow T \in SPT_{new}$ do
    if $Y \rightarrow T \in S$ then
        if not WaitingList($Y \rightarrow T$).contains(N) then
            //N already sent a CM for this upcoming link followNewSPT(T,N);
            if not CMSent.contains($Y \rightarrow T$) then
                SendToNeighbors(CM($Y \rightarrow T$));
                CMSent.add($X \rightarrow Y$);
            end
        end
        else
            //Do nothing, this part of the SPT will be followed
            //when $N$ sends the necessary completion message.
        end
    end
    else
        //This link is not an upcoming link, N sent the
        //necessary completion messages to continue the update
        //of the destinations behind this link.
        followNewSPT(T,N);
    end
end
```

Algorithm 3: Processing of a set of link metric decrease events

Waiting List becomes empty, R is allowed to perform its FIB update and send its own completion messages to all its neighbors.

During the whole convergence phase, when a packet reaches a router R that has updated its FIB, it is sure that the nexthop for its destination has sent a completion message to R, so that this nexthop has also updated its FIB. This property can be verified hop by hop along the path from R to X, so that the packet will reach X and will then be forwarded on a path containing routers whose paths towards the destination are not affected by the event.

5.5 Convergence time in ISP networks

In this section, we analyze by simulations the convergence time of the proposed technique, in the case of a link down event. The results obtained for link up events are very similar. Indeed, the updates that are performed in the FIB of each router for the shutdown of a link impact the same prefixes for the linkup of the link. The only difference in the case of a link up is that the routers do not need to compute a reverse Shortest Path Tree.

As no packets are lost during the convergence process, we cannot define the convergence time as the time required to bring the network back to a consistent forwarding state, as it would always be equal to zero. What is interesting to evaluate here is the time required by the mechanism to update the FIB of all the routers by respecting the ordering. A short convergence time is desired because other events occurring in the network during the ordered convergence process will force the routers to fall back to a fast, non loopfree, convergence, and we want to make this as rare as possible.

To perform this analysis, we took the measurements of [FFEB05] that presented the time to perform a SPT computation and a FIB update on current high-end routers. The ordering of the FIB update requires to compute the new Shortest Path Tree, and the computation of a reverse Shortest Path Tree in the case of a link down event. The Waiting List can be computed on the fly of the SPT computation, so that we only introduced a fixed amount of time to consider the computation of those lists.

We also added a fixed Hold Down before the process starts, in order to ensure that all the routers have received the link state packet describing the topology change before the scheme begins. We set the hold time before completion messages are being sent to 200 msec. This is a very large value compared to the time required to perform a SPT computation and a rSPT computation on the topologies under study. So, in our simulations, routers were ready to perform their FIB updates and send their completion messages when this hold time elapses.

Note that a router will start this Hold Down Timer as soon as it receives the Link State Packet describing the topology change. Thus, the time at which the Hold Down Timer expires on each router depends on the flooding time of link-state packets in the network. We also took the measurements of [FFEB05] to obtain the delay that is required to flood a link state packet from the router where the shutdown is performed towards the other routers in the network.

We assume that the time required to parse and process a Completion Message is similar to the time required to parse a Link-State Packet and insert it in the link-state database, i.e., a value between 2 msec and 4 msec [FFEB05]. When a router sends a completion message to a neighbor, it is thus removed from the neighbor's waiting list after the delay of the link on which the message is sent plus the time required to process a link-state packet. The time required to perform the FIB update in each rerouting router is obtained by computing their new FIB and multiplying the number of prefixes to update by the time to perform a prefix update that we obtained in the measurements (i.e., 100 μsec per prefix). The number of prefixes associated with each router is obtained from an IS-IS trace. A summary of the parameters of the simulation is presented in Table 5.1.

Table 5.1: Simulation parameters

lsp_process_delay	[2,4]ms
update_hold_down	200ms
(r)spf_computation_time	[20,30]ms in Tier-1 ISP
	[2,4]ms in GEANT
fib_prefix_update_delay	100 μs/prefix
completion_message_process_delay	[2,4]ms
completion_message_sending_delay	[2,4]ms

Our simulations work as follows. Upon an event, the link-state packet is flooded through the network. Upon reception of the link-state packet, each router starts its Hold Down Timer and computes its SPT, rSPT, and its Waiting List. When their Hold Down Timer expires, the routers that have an empty Waiting List perform their FIB update, and send their completion messages. When a router has finished the computation of its SPT and rSPT, it considers the completion messages that it has received. When a router has a non empty Waiting List, it waits for it to become empty, and then it performs its FIB update and sends its own completion message. For each link down event under study (link-id on the x-axis), we plot the time at which all the routers have updated their FIB, so that all the operations implied by the scheme have been performed. We sorted the link-ids according to the obtained convergence times.

Figure 5.4 shows the convergence times considering the removal of each directed link of Geant, an European research network containing 22 nodes and 72 (directed) links. We can see that, even if FIB updates are delayed, the convergence time remains short and the main component of the convergence is the fixed 200 msec hold time. The worst-case convergence time with the solution is 50 msec longer than the convergence time presented on the same topology in [FFEB05], when the same hold time is used.

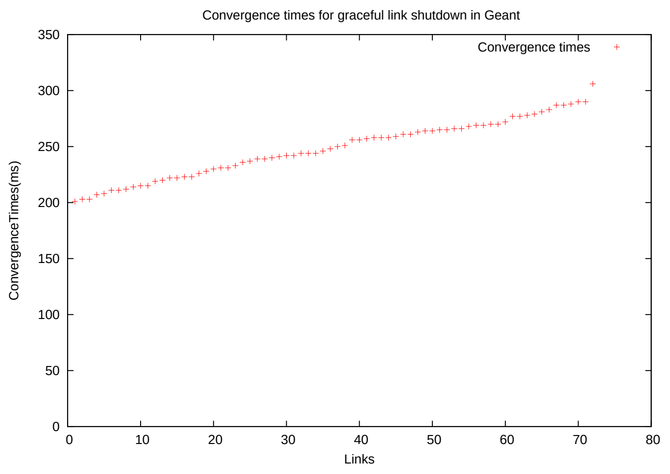


Figure 5.4: Convergence times in Geant

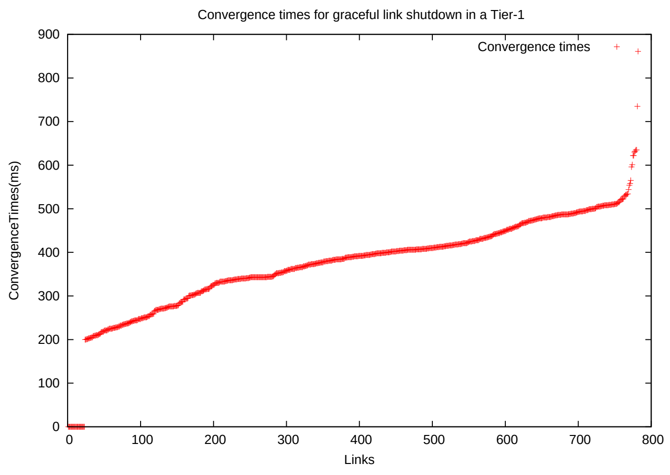


Figure 5.5: Convergence times in a Tier-1 ISP

Figure 5.5 shows the convergence times considering the removal of each directed link of a Tier-1 ISP. The values of 0 correspond to the shutdown of 23 directed links that did not carry packets due to their large IGP metric. This number is odd, which can be explained by the fact that some links have asymmetrical metrics, so that one direction of the link is used while the other not. The worst loop-free convergence time was 861 msec. This can be explained by the fact that the rSPT of this link contained a branch of 4 routers that had to perform a FIB update that lasted approximately 120 msec. The other components of the convergence are the 200 msec to compute the SPT and rSPT, and the delays of the links on which the completion messages were sent. Compared to [FFEB05] the convergence time is in the worst-case 400 msec longer than the convergence time when loops are not avoided.

To conclude, this analysis shows that a sub-second convergence is feasible even if a loop avoidance mechanism is used. The increase in the convergence time compared to the convergence time without the loop avoidance mechanism is small. With the solution operators could shut down links in their topology without loosing packets, by letting the network adapt to the change and stop using the link within one second, so that the use of the mechanism would not be a constraint for the operators.

In order to reduce the delaying of the FIB Updates as much as possible, we combined the proposed solution with a technique that lets a router find if its new nexthop for a destination already provides a loop-free path. So that, in some cases, routers can safely update their FIB for the destination without respecting the ordering. In the next section, we will briefly explain this technique, and we will evaluate the provided gain in the convergence time.

5.6 Ranking Shortcuts

As explained in the previous section, the motivation for shortcuts is to reduce as much as possible the delaying of the FIB updates, which is the interval between the moment at which a router is ready to update its FIB for a destination by using the nexthops corresponding to the new shortest paths through the network, and the moment at which the router actually does it.

In this section, we will show that a router applying the proposed ordering scheme will implicitely compute a sufficient information to decide wether it can shortcut the scheme and perform its FIB update directly, while preserving the transient forwarding consistency accross the network.

The decision to use this optimization is local to the router, i.e., each router can independently decide to apply the shortcut or not.

In the case of a link $X \rightarrow Y$ down or metric increase event, a router R computes $rSPT(X \rightarrow Y)$. From this tree, R obtains the set of routers that are using R to reach Y via $X \rightarrow Y$.

By doing this, R also computes the set of unaffected routers, i.e., the routers that do not use the link $X \rightarrow Y$ at all. These are the routers in $rSPT(Y)$ that do not have a path towards Y that contains $X \rightarrow Y$. Routers that are below $X \rightarrow Y$ can be marked during the computation of the rSPT, so that, at the end of the computation, a router N that is not marked is known to be an unaffected router, so that $X \rightarrow Y \notin SPT(N)$.

The shortest paths from this router to the destinations that R will have to reroute will not change, so that if the new nexthops of R for one destination belong to this set of unaffected routers, R is allowed to directly reroute the destination towards these new nexthops by disregarding its rank or the state of its Waiting List.

Several implementations of this shortcut are possible. Firstly, one router can decide to perform a full FIB update by shortcutting its rank if all the new nexthops to which it will reroute packets are unaffected routers. Secondly, a router can decide, destination per destination, if the set of new nexthops for one destination only contains unaffected routers. When this is done, the router is allowed to update its FIB for those destinations directly, and perform a second FIB update with the remaining destinations by respecting its rank or when its Waiting List becomes empty.

The first solution is the simplest, and preserves the property that routers update their FIB in one shot in the case of a single link event. The second solution is more complex, but this shortcut will be applicable more often.

To evaluate the gain of such shortcuts, we performed the same analysis as presented in Section 5.5, by considering the first shortcut solution. More precisely, when the Hold Down Timer expires in a router which is allowed to apply the shortcut, the router performs its FIB update directly. Note that this router will not send its completion message before its Waiting List is empty, in order not to change the meaning of a completion message. But, when a router has already performed its FIB update when its Waiting List becomes empty, it is allowed to send its own completion message directly.

In Geant, the gain was negligible. This can be explained by the fact that a small amount of prefixes are advertised in Geant, so that the FIB update time component is negligible compared to the Hold Down time, and the sending of completion messages through the network.

In the tier-1 ISP, the gain of the shortcut is more perceptible, because many prefixes are advertised in the network, and in many link maintenance cases, the rerouting routers were allowed to do the shortcut. For example, in the worst-case convergence time of 861 msec without shortcuts, the convergence time with shortcuts is 736 msec. In fact, some of the routers that were contributing to this long convergence time could safely perform their FIB updates in parallel.

We analyzed the coverage of both shortcut mechanisms, and found out that in the Tier-1 ISP, 54 % of the FIB updates that had to be performed by routers during the analysis could be shortcut with the first solution. With the second shortcut solution, 69 % of the FIB updates could be shortcut for at least one prefix. The second shortcut solution does not provide a significative gain in coverage. As the goal of the scheme was to permit an ordered convergence where, in the case of

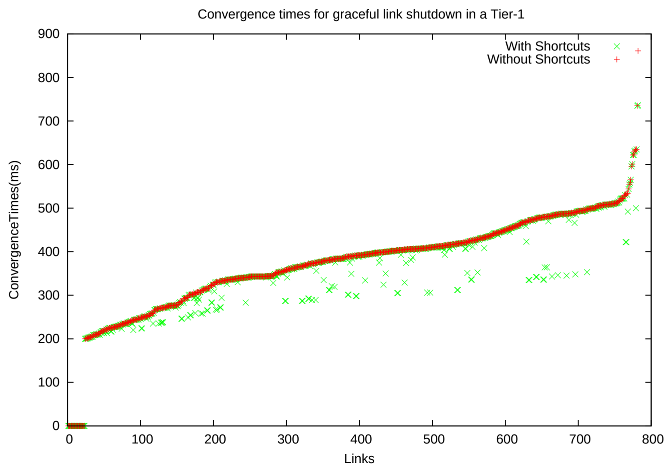


Figure 5.6: Convergence times in a Tier-1 ISP

a single link event, all the prefixes can be updated in one shot, we think that the first solution is to be preferred over the second. As the application of any shortcut solution can be decided independently by each router of the network, the choice of applying one method or another or not applying a shortcut at all can be made according to the software design and performance of each router of the network.

5.7 Related Work

The problem of avoiding transient loops during IGP convergence has rarely been studied in the literature although many authors have proposed solutions to provide loop-free routing. An existing approach to loop-free rerouting in a link-state IGP [GLA89] requires that the rerouting routers take care of routing consistency for each of their compromised destinations, separately. In fact, those mechanisms were inspired by distance-vector protocols providing a transiently loop-free convergence [JM82]. With this kind of approach, a router should ask and wait clearance from its neighbors for each destination for which it has to reroute. This implies a potentially large number of message exchanged between routers, when many destinations are impacted by the failure. Every time a router receives clearance from its neighbors for a given destination, it can only update forwarding information for this particular one. This solution would not fit well in a Tier-1 ISP topology where many destinations can be impacted by a single topological change. Indeed, in such networks, it is common to have a few thousands of prefixes advertised in the IGP [FFEB05]. Note that those solutions do not consider the problem of traffic loss in the case of a planned link shutdown.

In [SCK$^+$03], a new type of routing protocol allowing to improve the resilience of IP networks was proposed. This solution imposes some restrictions on the network topology and expensive computations on the routers. Moreover, they do not address the transient issues that occur during the convergence of their routing protocol. In [NST99], extensions to link-state routing protocols are proposed to distribute link state packets to a subset of the routers after a failure. This fastens the IGP convergence, but does not solve the transient routing problems and may cause suboptimal routing.

In [LYN+04], transient loops are avoided when possible by using distinct FIB states in each interface of the routers. Upon a link failure, the network does not converge to the shortest paths based on the new topology. Indeed, the failure is not reported. Instead, the routers adjacent to the failed link forward packets along alternate links, and other routers are prepared to forward packets arriving from an unusual interface in a consistent fashion towards the destination. As such, the solution is a Fast Reroute technique. Our solution is orthogonal to [LYN+04] as our goal is to let the network actually converge to its optimal forwarding state by avoiding transient forwarding loops when a Fast Reroute mechanism has been activated, or when the failure is planned.

In [ZKN+05], transient loops are avoided by selectively discarding the packets that are caught in a loop, during a fast convergence phase following an unplanned event. The idea is to also to use distinct FIB states in each interface of the routers, and let routers drop packets when they would be caught in a loop. Care has been taken to avoid dropping a packet arriving from an unusual interface if the router cannot ensure that the packet is actually caught in a loop. Once again, our goals differ as we focus on transient loops occuring during the convergence from an initial forwarding state to the optimal forwarding state based on the new topology.

The problem of gracefully changing the network topology without disrupting traffic has been addressed in MPLS networks using traffic engineered tunnels. In these networks, RSVP-TE [ABG+01] is used to create and modify the MPLS tunnels between an ingress and an egress router. When a traffic engineered tunnel must be modified, for example to follow a different path, RSVP-TE allows to change the tunnel without loosing any packet.

5.8 Applicability of the solution

In this section, we discuss the applicability of the solution w.r.t. the routing system to which it is intended to be applied and the type of topological change it is able to handle.

The scheme proposed in this chapter can be applied for any shortest path routing system whose convergence process after a topological change respects the two following properties. First, the topological change is made of a set of link metric increase and/or link shutdown events, or the topological change is made of a set of link metric decrease and/or link up events. Second, the routing system must be such that a node of the system can directly decide, based on its current knowledge of the topology and its knowledge of the topology change, the set of FIB updates that it will perform to reach its post-convergence forwarding state. This second property typically means that the solution is applicable for link-state routing protocols where nodes have a complete knowledge of the topology.

Loops can be avoided in the system by applying the scheme, disregarding the nature of the event leading to the change.

When the event is predictable, e.g., a maintenance operation, applying the scheme always improves the convergence as the only source of packet losses comes from transient forwarding loops. When the event is sudden, e.g., a fiber cut, and the affected links are protected with a Fast Reroute scheme, applying the scheme always improves the convergence as the only source of packet losses after the activation of the protection also comes from transient forwarding loops.

When the event is sudden and some of the affected links are not protected with a Fast Reroute scheme, applying the solution introduces a trade-off between packet loss avoidance and loop avoidance. Indeed, the solution relies on a delaying of some FIB updates by some nodes of the network. Thus, the scheme delays the restoration of paths around the failed components when applied in such cases, and more packets could be last than if nothing had been done to prevent forwarding loops from occuring. Applying the scheme in such cases could be justified for example when the performance of the updates of the FIB can be hampered by the occurance of forwarding loops to such an extent that the restoration of paths around the failed components would be slower when forwarding loops are not avoided. Considering the router hardware and software used in Today's large ISP networks, we currently do not recommend to apply the scheme in such scenarios.

5.9 Conclusion

The first important contribution of this chapter is that we have proved that it is possible to define an ordering on the updates of the FIBs that protects the network from transient loops. We have proposed an ordering applicable for the failures of protected links and the increase of a link metric and another ordering for the establishment of a new link or the decrease of a link metric. We also proposed orderings that are applicable in the case of a non-urgent router down or up event, as well as line card events. Then, we generalized the scheme to events affecting any kind of sets of links in the network. Next, we presented optimizations to the scheme that allow routers to update their FIB by disregarding the proposed ordering when it is proved not to lead to forwarding loops.

Finally, we have shown by simulations that our loop-free extension to currently deployed link-state protocols can achieve sub-second convergence in a large Tier-1 ISP.

Chapter 6

Forwarding Loop avoidance using link metric reconfigurations

In the previous chapter, we described modifications to link-state IGP to avoid microloops during the convergence. What we propose in this chapter is a loop avoidance technique that does not require modifications to IS-IS and OSPF, and that can be applied **now** by ISPs when a **topology reconfiguration** has to be performed. Roughly, in the case of a manual modification of the state of a link, we progressively change the metric associated with this link to reach the required modification by ensuring that each step of the progression will be loop-free. The number of changes that are applied to a link to reach the targeted state is minimized. Analysis performed on real regional and tier-1 ISP topologies show that the number of required transient changes is small. The solution can be applied in the case of link metric updates, manual set up, and shut down of links.

6.1 Introduction

This chapter presents a solution relying on progressive reconfigurations of a link metric such that the desired updated state of the link can be reached by never putting the routers of the network in an inconsistent forwarding state during the convergence process. In essence, the solution does not require modifications to the routing protocols or router software, as changing a link metric has always been a feature of Link-State Interior Gateway Protocols.

The chapter is organized as follows. We firstly illustrate the problem and the solution with a small example. In section 6.2 we introduce a few notations and the basic properties on which the proposed solution relies, and we prove that there always exists a sequence of metrics that permits to reach the desired link-state without introducing transient forwarding loops. In Section 6.3, we present how to compute short metric sequences that can be used to adapt to a metric increase or the removal of a link by avoiding transient forwarding loops. In Section 6.4, we present the solution for the case of a link metric decrease and a link reactivation.

In Section 6.5, we present the results of an analysis performed on ISP topologies, showing that the Merged Reroute Metric Sequences are short in practice. In Section 6.7, we present the related work, and we conclude the work in section 6.8.

6.2 Loop free convergence using metric increments

Let us illustrate the transient routing loops mentioned in chapter 4, with Figure 6.1. In this network composed of five routers and six links, all links have an IGP metric of 1 except the link between routers A and B whose IGP metric is set to 5. Let us consider what happens when link $B-C$ needs to be shutdown for maintenance reasons. This link can be shutdown in one step, by removing it from the link state database or in two steps as proposed in [TR06] by first setting its IGP metric to $MAX\_METRIC-1$ and later removing it from the link state database. In both cases, after the first step all routers must update their FIB. Before the topology change, router B sent the packets towards A via C. After the topology change, it will send the packets via D. Unfortunately, before the topology change, router D was sending the packets towards A via routers B and E. This implies that if router B updates its FIB before router D, a likely event as router B will learn the topology change before router D, then packets destined to A will loop on the $B-D$ link until router D has updated its FIB.

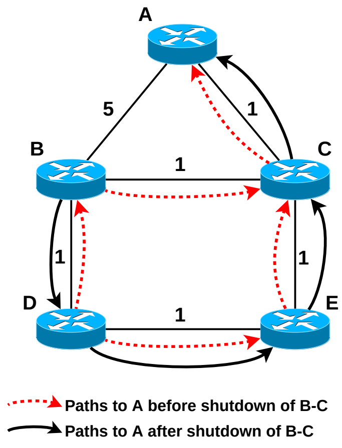

A
B
C
D
E
5
1
1
1
1
1
Paths to A before shutdown of B-C
Paths to A after shutdown of B-C

Figure 6.1: Simple network

Let us reconsider the example above, we will see that there exists a sequence of metrics for link $B-C$ that permits to shut down the link without causing packet loops and losses. Next, we will show that, in any possible network topology, there always exists a sequence of metric increments that will allow a loopfree convergence for the metric update of a link $A \rightarrow B$ from one value m to another $m' > m$.

Let us assume that the IGP metric of link $B-C$ changes from 1 to 2 in the

Router	A	B	C	D	E
A	-	C	C	C	C
B	C	-	C	D	C and D
C	A	B	-	B and E	E
D	E and B	B	E and B	-	E
E	C	C and D	C	D	-

Table 6.1: FIB of all routers when $B - C = 1$

Router	A	B	C	D	E
A	-	C	C	C	C
B	C	-	C	D	~~C and~~ D
C	A	B	-	~~B and~~ E	E
D	E ~~and B~~	B	E ~~and B~~	-	E
E	C	~~C and~~ D	C	D	-

Table 6.2: FIB of all routers when $B - C = 2$

Router	A	B	C	D	E
A	-	C	C	C	C
B	~~C~~ D	-	~~C~~ D	D	D
C	A	~~B~~ E	-	E	E
D	E	B	E	-	E
E	C	D	C	D	-

Table 6.3: FIB of all routers when $B - C = 4$

topology of Figure 6.1. Before the change, the FIB of all routers is as shown in table 6.1. When the metric of link $B - C$ is set to 2 (in both directions), routers B, C, D and E update their FIB. At router B, the consequence of the metric change is that it will stop using router C to reach destination E. C will stop using B to reach D, and D will stop using B to reach C and A. Thus, the metric change has reduced the number of equal cost paths used by some routers to reach several destinations. It is interesting to note that no transient loops occur during this metric change.

Let us look at what happens when the metric of link $B - C$ changes from 2 to 4. The new FIB of all routers is shown in table 6.3. This change caused routers B and C to update their FIB. Routers B and C no longer use link $B - C$ to reach any destination. As in the previous step, there are no transient loops during this update and with this metric value, link $B - C$ does not carry packets anymore. It can thus be safely shut down by the operator.

Now, let us show that metric sequences allowing a loopfree convergence always exist. We firstly introduce a few notations. $SPT_{A \underset{m}{\rightarrow} B}(X)$ is the shortest path tree of X based on the initial topology where the metric of the link $A \rightarrow B$ is set to

m[1]. $Paths(X, Y, S)$ is the set of equal cost paths from X to Y in the shortest path tree S. $Dist(X, Y, S)$ is the IGP distance from X to Y according to the shortest path tree S. When a change in a link metric is performed, we use $Dist(X, Y)$ to denote the distance from X to Y before the change, and $Dist'(X, Y)$ to denote the distance from X to Y after the change. $rSPT(X)$ is the reverse Shortest Path Tree of X. This is a tree containing all the shortest paths from the nodes of the network graph towards X. Note that when Equal Cost Paths are used, this graph is actually an acyclic graph. When a change in a link metric is performed, we respectively denote the rSPT of X before and after the change with $rSPT(X)$ and $rSPT'(X)$.

Definition 6.2.1 *We say that a change is loopfree for a destination D if transient forwarding loops during the routing convergence cannot occur. That is, there does not exist an ordering of the FIB updates for destination D that transiently puts the network in an inconsistent forwarding state such that packets destined to D can loop.*

Theorem 6.2.2 *A change is loopfree for destination D if and only if the merging of $rSPT(D)$ with $rSPT'(D)$ does not contain a cycle.*

Theorem 6.2.2 has been proved in theorem 4.1.1.

Definition 6.2.3 *A change is loopfree if it is loopfree for all the nodes of the network.*

To prove the existence of a sequence of metric increments that allows a loopfree convergence when updating the metric of a link, we will show that incrementing the metric of the link by 1 never causes transient loops, so that progressively incrementing the metric of a link can be performed to avoid loops.

Theorem 6.2.4 *In a stable network, incrementing the metric of a link $A \longrightarrow B$ by one leads to a loop-free convergence process.*

We can prove this theorem by contradiction. Let us show that it is absurd to have a transient loop in the network when the metric of link $A \rightarrow B$ is increased by one. There can be a loop for a destination D while the routers adapt to the metric change if there exists two distinct nodes X and Y such that X was in the paths from Y to D before the change, and Y will be in the paths from X to D after the change. In other words, there can be a transient loop for packets destined to D if the merging of the rSPT of D before and its rSPT after the change contains a cycle.

$$X \in Paths(Y, D, SPT_{A \underset{m}{\longrightarrow} B}(Y)) \tag{6.1}$$
$$Y \in Paths(X, D, SPT_{A \underset{m+1}{\longrightarrow} B}(X)) \tag{6.2}$$

[1]Although the use of Equal Cost Multi Path makes this "tree" actually be an acyclic graph, with potentially more than one shortest path from a source to a destination, we use the term "tree" to respect the IS-IS and OSPF terminology.

If X was in the paths from Y to D before the change, X was not using Y to reach D before the change, so that if (6.2) is true, then the new SPT of X is such that one of the shortest paths from X to D contains Y and its length is the length of its initial shortest path to D plus 1 :

$$\begin{aligned} & Dist'(X,Y) + Dist'(Y,D) \\ = \quad & Dist(X,D) + 1 \end{aligned} \tag{6.3}$$

If Y was using X to reach D before the change, then

$$Dist(Y,D) = Dist(Y,X) + Dist(X,D) \tag{6.4}$$

In a **first case**, when $Dist(Y,D) = Dist'(Y,D)$, by replacing $Dist'(Y,D)$ in (6.3) by the value of $Dist(Y,D)$ in (6.4), we obtain

$$\begin{aligned} & Dist'(X,Y) + Dist(Y,X) + Dist(X,D) \\ & = Dist(X,D) + 1 \end{aligned} \tag{6.5}$$

Thus,

$$Dist'(X,Y) + Dist(Y,X) = 1 \tag{6.6}$$

Which is impossible as X and Y are two distinct nodes and the sum of two path lengths must at least be equal to 2.

In the **other cases**, Dist'(Y,D) is equal to Dist(Y,D)+1, as only one metric of a link has been updated by incrementing it by 1. By replacing $Dist'(Y,D)$ in (6.3) by the value of $Dist(Y,D)$ in (6.4) plus one, we obtain

$$\begin{aligned} & Dist'(X,Y) + Dist(Y,X) + Dist(X,D) + 1 \\ & = Dist(X,D) + 1 \end{aligned} \tag{6.7}$$

From 6.7, we obtain

$$Dist'(X,Y) + Dist(Y,X) = 0 \tag{6.8}$$

Which is impossible as X and Y are two distinct nodes. Thus, it is impossible to increment a link metric by one and verify both (6.1) and (6.2), which are necessary for a transient forwarding loop to happen. ∎

We have thus proved that we can always change the metric of a link to a larger metric, by progressively incrementing the metric of the link by one, until the target metric is reached. When the link must be shut down, the metric can be incremented until it becomes so large that the link does not carry packets anymore. When this metric has been reached, the link can be safely shut down.

6.3 Loop free convergence using Key metric increments

Unfortunately, the technique described above is **inefficient** as a large number of increments could have to be used when a link with a low metric must be shut down. To solve this problem, we propose to perform larger increments of the metrics when they are known to lead to a loopfree convergence. As the metric space of links is wide in IS-IS and OSPF, it is not realistic to totally explore the metric space and try to find a possible loop free increment sequence for a given link metric transition. Indeed, many operators take advantage of the whole width of the metric space. For example, in the European Geant Research Network [GEA], there exists a link with a metric of 1 and a link with a metric of $20,000$. Such variety of link metrics is also present in the tier-1 ISP topologies that we analyse in Section 6.5.

Let us consider the topology of Figure 6.2. If we were to set the metric of the link $B-C$ to 40 with the previous technique, we would have to perform 30 metric changes.

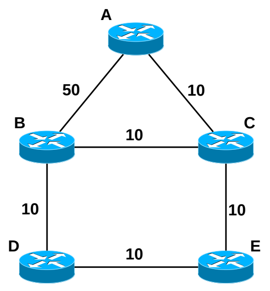


Figure 6.2: Simple network with large metrics

However, we can see that even though the metric transition for link $B-C$ from 10 to 40 leads to a forwarding loop, the transition from 11 to 40 could not cause a forwarding loop, so that $\{10, 11, 40\}$ is a valid metric sequence to change the metric of the link without loosing packets.

Now, we identify several key aspects of the transition from one link metric to another, that we will use to reduce the set of metric increments used to perform a progressive loopfree convergence.

6.3.1 Reroute Metric Sequences

Let us consider the set of equal cost shortest paths from a source S towards a destination D, such that some of these paths contain a link $A \rightarrow B$. We can identify three different cases when the metric of this link is incremented by 1.

The first case is when the metric increase **does not change the forwarding path from S to D**; except that the new distance from S to D is increased by one. In this case the set of paths from S to D does not change. This implies that all the paths used by S to reach D before the change contained the link $A \rightarrow B$.

Indeed, if this was not the case only the paths that do not contain this link would be used after the change, as their length is not affected. Note that in this first case $Dist'(S,D) = Dist(S,D) + 1$. For example, when the metric of link $B \rightarrow C$ in Figure 6.2 is changed from 10 to 11, the paths from B to C do not change, and the distance between B and C is increased by 1.

The second case is when the metric change **increases** the number of equal cost paths from S to D. This is the case when the paths via the link $A \rightarrow B$ are still among the shortest paths towards D after the change, and other paths to D not via $A \rightarrow B$ now become shortest paths. Note that in this case, $Dist'(S,D) = Dist(S,D) + 1$. For example, when the metric of link $B \rightarrow C$ in Figure 6.2 is changed from 29 to 30, the previous paths from B to C are still used, and another path via D and E is used.

The third case is when the metric change **decreases** the number of equal cost paths from S to D. This is the case when equal cost paths to D, not via $A \rightarrow B$, existed before the change, and are the sole paths being used by S after the change. In this case, $Dist'(S,D) = Dist(S,D)$. For example, when the metric of link $B \rightarrow C$ in Figure 6.2 is changed from 30 to 31, only the path $B \rightarrow D \rightarrow E \rightarrow C$ is used by B to reach C.

Keeping this in mind, let us focus on a particular ordered sequence of metrics for a link $A \rightarrow B$, considering an initial metric m_1, a target metric m_t, and a destination D initially reached via this link by some routers. This sequence, called "Key Metric Sequence" (KMS), contains m_1, m_t, and all the metrics within $[m_1, m_t]$ for the link $A \rightarrow B$ that will force at least one router R to use an additional equal cost path towards D that does not contain $A \rightarrow B$. We will call m the "Key Metric" for destination D at R if R uses an additional path not via $A \rightarrow B$ when the link metric is set to m.

In Figure 6.2, the Key Metric Sequence for link $B \rightarrow C$, considering an initial metric of 10, a target metric of 40, and destination A is $\{10, 30, 40\}$. 30 is the Key Metric for destination A at node B since B will start using path $B \rightarrow D \rightarrow E \rightarrow C$ to reach A when the metric is set to 30. 10 is the initial metric, and it is also the Key Metric for destination A at node D since D uses both paths via and not via $B \rightarrow C$ to reach A when the metric of the link is 10.

Computing the KMS of a destination D, considering a link $A \rightarrow B$, its initial metric m_i, and a target metric m_t for this link is simple. We compute the rSPT of D with both initial and target metric for $A \rightarrow B$. When the distance from a node N to D differs in those rSPTs, $m_i + Dist'(N,D) - Dist(N,D)$ is inserted in the sequence. This metric is the one that will let N use paths via as well as not via $A \rightarrow B$ to reach D, so that this value is the Key Metric of N.

Let us consider one KMS $\{m_1, m_2, \ldots, m_i, \ldots, m_t\}$ for a destination D. Let us now insert, between each pair of elements (m_i, m_{i+1}), an intermediate value m'_i equal to $m_i + 1$.

We will show in Theorem 6.3.1 that such a sequence, that we call a Reroute Metric Sequence (RMS) for destination D, is such that the progressive setting of each metric contained in the sequence provides a loop free convergence for D, for each successive metrics in the sequence, until the target metric is reached.

In Figure 6.2, the Reroute Metric Sequence for link $B \rightarrow C$, considering an inital metric of 10, a target metric of 40, and destination A, is $\{10, 11, 30, 31, 40\}$. If the metric of the link is progressively set to those values, then no transient forwarding loop could occur for destination A.

Theorem 6.3.1 *Given a link $A \rightarrow B$, progressively setting the metric of the link with the metrics of a Reroute Metric Sequence for D will provide a loop free convergence for destination D.*

Let us consider a RMS for a link $A \rightarrow B$ and a destination D, $\{m_1, m_1 + 1, m_2, m_2 + 1, \dots, m_i, m_i + 1, \dots, m_t\}$.

For each i, a transition from m_i to $m_i + 1$ is loopfree according to Theorem 6.2.4.

For each i, a transition from $m_i + 1$ to m_{i+1} is loopfree. In a first case, if $m_{i+1} = m_i + 1$ there is no metric increment to perform. Otherwise, if the metric of $A \rightarrow B$ is $m_i + 1$, there is no router that will update its FIB for destination D if the metric of the link is set to a value within $[m_i + 1, m_{i+1}[$. The contrary would mean that there is a rerouting router whose Key Metric is not present in the RMS. So, increasing the metric of the link from $m_i + 1$ to m_{i+1} is equivalent to changing the metric of the link from $m_i + 1$ to $m_{i+1} - 1$, which does not change anything in the paths used by the routers to reach D, and then incrementing the metric of the link from $m_{i+1} - 1$ to m_{i+1}. Doing this cannot cause forwarding loops according to Theorem 6.2.4. ▮

We showed in the beginning of this section that, in the topology depicted in Figure 6.2, the Metric Sequence $\{10, 11, 40\}$ was sufficient to provide a loopfree convergence for destination A when setting the link metric of $B - C$ to 40, even if the RMS computed for this link would have been equal to $\{10, 11, 30, 31, 40\}$ for A.

In fact, most of the metrics of a RMS are actually not necessary to provide a loopfree convergence for a given destination D. But these are the key metrics that cause FIB Updates for destination D on the routers of the network. So, we will try to remove the unnecessary increments from the RMS. We will call the obtained sequences Reduced Reroute Metric Sequences (RRMS). When the size of a RRMS for a destination D is minimal, i.e. when there does not exist a shorter metric sequence ensuring a loop-free convergence, we call the sequence an Optimal Reroute Metric Sequence (ORMS).

6.3.2 Reduced and Optimal Reroute Metric Sequences.

Definition 6.3.2 *An optimal reroute metric sequence, given a topology change, is a loopfree reroute metric sequence for this change that is shorter or of equal length than any other loopfree reroute metric sequence for this change.*

Here, we will explain our technique to reduce an RMS to an RRMS, considering a destination D, a link $A \rightarrow B$, with its intial metric m_1 and a target metric

$m_t > m_1$. Next, we will prove that our technique provides Optimal Reroute Metric Sequences.

To reduce a RMS for a destination D to an RRMS, we propose to start from the initial metric and perform the largest possible metric increment that does not lead to forwarding loops. We do that at each step until the target metric is reached. We call this technique the "Largest Increase First" technique (LIF).

For example, given a Reroute Metric Sequence $\{m_1, m'_1, m_2, m'_2, \ldots, m_i, m'_i, \ldots, m_t\}$, we find the largest metric M in that sequence, such that setting the metric of $A \rightarrow B$ to M will not lead to forwarding loops. To do that, we compute the rSPT of D considering the largest metric for the link in the sequence. Then, we merge the initial rSPT of D with its rSPT after the change, and we detect cycles within the obtained graph. When a cycle is detected, we try it again with smaller metrics until we find one metric M such that the merging of the rSPTs is cycle free. Then we reapply the technique, starting from M, and we do that repeatedly until we reach the target metric m_t.

When computing the largest metric increment, we chose to try the largest metric first and decrease it when cycles are detected to be able to reuse the rSPTs computed with large metrics during the remainder of the RMS reduction. Also, very few metrics are generally necessary to reach the target metric even if the initial RMS is long. Thus starting by the end of the sequence reduces the number of rSPTs to compute during the RMS reduction.

Theorem 6.3.3 *The reduction technique above provides optimal reroute metric sequences.*

Now, let us prove theorem 6.3.3. The reasoning is based on lemma 6.3.4.

Proof :

Lemma 6.3.4 *If a metric transition for a link $A \rightarrow B$ from m to n, with $m < n$, is not loopfree, then*

1. *A metric transition from k to n for this link, with $k < m$, is not loopfree*
2. *A metric transition from m to o for this link, with $n < o$, is not loopfree*

Let us prove this lemma. If the transition from metric m to n is not loopfree for a destination D, then there is a cycle in the merging of $rSPT(D)$ and $rSPT'(D)$, being respectively the rSPT of D when the metric of $A \rightarrow B$ is set to m and n.

Let us denote the rSPT of D when the metric of the link is set to o with $rSPT''(D)$. The second proposition is true if there is a cycle in the merging of $rSPT(D)$ and $rSPT''(D)$. When setting the link metric from m to n, the shortest path of a set of nodes towards D were no longer via link $A \rightarrow B$, which led to the possibility of a loop. Let us denote this set of nodes by $\mathcal{N}$. If the link metric was set to o, instead of being set to m, each node in $\mathcal{N}$ would also use their shortest paths to D not via $A \rightarrow B$. Basically, these are the same as the ones they use when the metric is set to n. So, the path from each node in $\mathcal{N}$ to D in $rSPT'(D)$ is the same path as in $rSPT''(D)$. So, when merging $rSPT''(D)$ with $rSPT(D)$, we obtain at least the same cycles as when merging $rSPT(D)$ with $rSPT'(D)$.

The same reasoning can be applied to prove the first proposition. ∎

From this lemma, we can prove that our reduction technique provides Optimal Reroute Metric Sequences.

Let us consider a RRMS $\{m_1, \ldots, m, m'', m''', \ldots, m_t\}$, obtained with our technique.

Due to the definition of the LIF technique, we know that

1. A transition from m'' to the metric of the initial RMS following m''', say m^{loopy} is not loopfree.

2. From 1) and Lemma 6.3.4, we know that a transition from a metric $x < m''$ to m^{loopy} is not loopfree.

If the LIF technique does not always provide an ORMS, this implies that another technique could provide a shorter valid sequence by not always selecting as next metric to a given m the largest possible metric increment that ensures a loopfree convergence. Starting from metric m, the better technique would thus select as next metric in its resulting sequence a metric $m' < m''$.

3. In order to spare a metric increment in comparison with LIF, it would have to select as the next metric after m', a metric $m^{better} > m'''$, so that $m^{better} \geq m^{loopy}$.

So, the better technique would have the subsequence $\{m, m', m^{better}\}$ in its Reroute Metric Sequence.

4. Knowing that $m' < m''$ and $m^{better} \geq m^{loopy}$, we obtain from 2) and Lemma 6.3.4 that this transition is not loopfree, so that this better technique does not exist. ∎

In Figure 6.2, the RMS for destination A, considering the metric change of link $B \rightarrow C$ from 10 to 40, is $\{10, 11, 30, 31, 40\}$. When applying the LIF technique the obtained ORMS for A is $\{10, 11, 40\}$. Indeed, a direct change from 10 to 30 would cause a loop between B and D, so that the metric 11 is mandatory, and a direct change from 11 to 40 is loopfree for destination A, so that the intermediate metrics are skipped by the technique.

6.3.3 Merged Reroute Metric Sequences.

In practice, routers react to the update of a link metric by updating their FIB for all the destinations towards which their shortest paths have changed. So, knowing the ORMS for a destination D, according to a metric transition for a link, is not sufficient to provide a working solution.

In this part, we show that the merging of the ORMS obtained for each destination gives a valid, loopfree Reroute Metric Sequence for all the destinations affected by the change. We call such sequences Merged Reroute Metric Sequences (MRMS).

Let us consider an ORMS $\{m_i, \ldots, m_j, m_k, m_l, \ldots, m_t\}$, for link $A \rightarrow B$ and a destination D.

We need to prove that inserting values in that sequence also gives a loopfree Metric Sequence for destination D.

Let us consider the sequence $\{m_i, \ldots, m_j, m_s, m_k, m_l, \ldots, m_t\}$, with $m_s \in]m_j, m_k[$. Let us denote the rSPT of D when the metric of link $A \rightarrow B$ is set to m_x by $rSPT_x(D)$.

As m_j and m_k are consecutive metrics in the initial ORMS, we know that the merging of $rSPT_j(D)$ and $rSPT_k(D)$ does not contain a cycle. The set of source-destination paths that differs between those rSPTs forms a superset of the paths that differ between $rSPT_j(D)$ and $rSPT_s(D)$. Indeed, every path not via $A \rightarrow B$ that becomes used to reach D when the metric of the link is set to m_s also becomes used when the metric of the link is set to a larger value. Also, every path via $A \rightarrow B$ that is still used to reach D when the metric of the link is set to m_k is also still used when the metric is set to $m_s < m_k$. This implies that the merging of $rSPT_j(D)$ and $rSPT_s(D)$ is the merging of $rSPT_j(D)$ and a subgraph of $rSPT_k(D)$, so that this merging does not contain a cycle. The same reasoning can be used to show that the merging of $rSPT_s(D)$ and $rSPT_k(D)$ is cycle free, so that the metric sequence $\{m_j, m_s, m_k\}$ is loopfree for destination D.

As the same reasoning can be applied when inserting a metric between m_s and m_k in the new sequence, we have proved that the insertion of an arbitrary number of metrics within an ORMS still gives a loopfree metric sequence for its destination. ∎

6.3.4 Optimization of Merged Reroute Metric Sequences.

The merging of two Optimal Reroute Metric Sequences S_a and S_b associated with two destinations a and b might be such that there exists a shorter sequence providing a loopfree convergence for both destination a and b.

Firstly, an Intermediate Metric in a Reroute Metric Sequence for S_a becomes unnecessary in the merged sequence if a Key Metric of S_b can play the role of the Intermediate Metric in S_a.

Let us for example assume that $S_a = \{3, 4, 8\}$, and $S_b = \{5, 8\}$, with 3, 8, and 5 being Key Metrics. The metric 4 in S_a is an Intermediate Metric introduced when the Reroute Metric Sequence is computed for a. This means that the only reason to transiently set the metric of the link to 4 is to force a router R to stop using its equal cost paths to a that contain $A \rightarrow B$, as 4 is not a Key Metric and the next Key Metric is 8. An intermediate value of 5 would have the same effect and would also be loopfree.

This implies that, $S'_a = \{3, 5, 8\}$ is also a valid Reroute Metric Sequence for destination a. So, we can replace the initial Merged Reroute Metric Sequence $\{3, 4, 5, 8\}$ by $\{3, 5, 8\}$, still ensuring that no transient forwarding loop will occur during the convergence.

Secondly, a Key Metric in a Reroute Metric Sequence for S_a becomes unnecessary in the merged sequence if another Metric present in S_b can play the role of this Key Metric. Let us for example assume that $S_a = \{3, 4, 8\}$, and $S_b = \{5, 8\}$, with 3,4,5,8 being Key Metrics. It is possible that the Key Metric 5 for S_b, obtained with the LIF technique, would be also valid if 5 is replaced by 4, so that $\{3, 4, 8\}$ would still ensure that no transient forwarding loops occur during the convergence.

To re-optimize the Merged Reroute Metric Sequences, we re-apply the LIF technique on them.

```
Metric increase to $m_t$ for Link $A \rightarrow B$:
//Computation of the affected Destinations
AffectedDest = follow($A \rightarrow B, SPT_{init}(A)$);
//Computation of the ORMS
ORMSSet = $\{\}$;
foreach Destination $D \in AffectedDest$ do
    RMS = GetRMS(D,$A \rightarrow B, m_t$);
    ORMS = OptimizeRMS($D, RMS, A \rightarrow B, l.metric, m_t$);
    ORMSSet.add(ORMS);
end
MergedRMS = MergeSequences(ORMSSet);
MergedRMS = PruneUnecessaryMetrics(MergedRMS);
return MergedRMS;

MetricSequence GetRMS(Destination dest, Link L, Metric target_metric):
RMS = $\{L.metric, target\_metric\}$;
//Compute the rSPT of D with the initial metric of L
initialRSPT = computeRSPT(dest,L,L.metric);
//Compute the rSPT of D with the target metric of L
targetRSPT = computeRSPT(dest,L,target_metric);
foreach Node
$S \mid PathLength(S, D, initialRSPT) \neq PathLength(S, D, targetRSPT)$ do
    KeyMetric = L.metric + PathLength(S,D,targetRSPT) -
    PathLength(S,D,initialRSPT);
    RMS.add(KeyMetric);
    //Introduce Intermediate Metric
    if KeyMetric $\neq$ target_metric then
        RMS.add(KeyMetric+1);
    end
end
return RMS;
```

Algorithm 4: Algorithm to compute Merged Reroute Metric Sequences. Part 1

In Figures 4 and 5, we present the pseudo-code for the computation of a Merged Reroute Metric Sequence considering a metric increase to m_t for a link $A \rightarrow B$.

```
MetricSequence **OptimizeRMS**(Destination $D$,MetricSequence RMS,Link L,Metric
StartMetric,Metric TargetMetric):
tempORMS = $\{StartMetric\}$;
currentMetric = StartMetric;
**while** *(!(currentMetric==TargetMetric))* **do**
    //Find the largest Metric M in RMS such that transition from
    //currentMetric to M is loopfree for destination $D$
    M = TargetMetric;
    bool loopfree=false;
    **while** *(! loopfree)* **do**
        MergedrSPT = merge(rSPT(D,L,currentMetric),rSPT(D,L,M)):
        **if** *MergedrSPT.containsCycle()* **then**
            M = Metric Before $M$ in RMS;
        **end**
        **else**
            loopfree = true;
        **end**
    **end**
    tempORMS.add(M);
    CurrentMetric = M;
**end**
**return** *tempORMS*;

ShortestPathTree rSPT(Destination Dest, Link L, metric m):
**if** *(rSPTCache.contains(Dest,L,m))* **then**
    **return** *getrSPTCache(Dest,L,m)*
**end**
**else**
    rSPT = Compute rSPT of D with the metric of L set to m;
    putInCache(Dest,L,m,rSPT);
**end**
```

Algorithm 5: Algorithm to compute Merged Reroute Metric Sequences. Part 2

The algorithm firstly explores the SPT of A to obtain the set of destinations that are reached via link $A \rightarrow B$. Then, it computes the Optimal Reroute Metric Sequence for each destination D reached via this link. To do that, it computes the set of Key Metrics for D, by analysing the reverse Shortest Path Trees of D with the initial and target metric set to $A \rightarrow B$, and it inserts the Intermediate Metrics to give the Reroute Metric Sequence.

Then, it optimizes the Sequences by applying the LIF technique. In the implementation, we stop the merging of the rSPTs performed by the LIF technique as soon as a length-2 cycle is detected, so that the cycle detection performed on the merged rSPTs is not necessary in those cases.

Finally, we merge the obtained optimal Reroute Metric Sequences, and we prune Intermediate and Key Metrics that become unnecessary due to the merging. Note that the computed rSPTs are put in a cache along the computation of an optimized reroute metric sequences, so that the number of rSPT computations is in the worst case equal to the length of the initial Reroute Metric Sequence for each destination.

The algorithm has been implemented in Java as a Proof of Concept.

6.4 Loop free convergence using metric decrements

What has been presented in the previous section holds for the cases where a link is shut down or its metric is increased. We based the correctness of the provided metric update sequences on the fact that, at each step, for each destination affected by the change, the merging of its rSPT before and after the event is cycle free.

That is, when we consider the transition between a metric m_i towards a metric m_t smaller than m_i, we know that reversing a valid Reroute Metric Sequence for the transition of the link metric from m_t to m_i will provide transitions such that the merging of the rSPTs of the affected destinations are cycle free, at each step of a transition from m_i to m_t.

So, it is not necessary to provide an algorithm that specifically solves the metric decrease problem as soon as an algorithm is provided for the metric increase problem.

Note that when a link is being brought up in the network, we first set the metric of the link to a value such that the link will not be used. Then, we apply the same technique as for a metric decrease event.

6.5 ISP Topologies Analysis

To evaluate the performance of our rerouting scheme, we use three real ISP topologies. The first one is GEANT, the pan-European Research Network [GEA]. We use the GEANT topology as it was in 2005. GEANT connected all the National Research networks in Europe and had interconnections with research networks in other continents. GEANT was composed of 22 routers, 21 in Europe and one in New-York, USA. The network topology was highly meshed in the core (Germany, Switzerland, France, UK, Netherlands) and there was fewer redundancy in the other parts of the network. Each POP was composed of a single router.

The second studied network contains all the routers of a Tier-1 ISP with presence in Europe, America and Asia. This network is composed of about 110 routers and 400 directed links.

The third studied network contains the backbone nodes of a large Tier-1 ISP. The backbone of this network has about 200 routers and 800 directed links in Europe, America and Asia. For both Tier-1 ISPs, each POP is usually composed of two core routers as well as several aggregation and access routers.

We applied the technique on all the directed links of those ISPs. We did not try to write optimized Java code in our proof of concept. However, the time required to compute the reroute metric sequences for Geant was negligible. For the two Tier-1 ISPs, a few seconds was required in the worst case to compute a reroute metric sequence. As we will see in the results, around 50% of the links shutdown could lead to a forwarding loop in the studied topologies. So, directly setting the metric of a link to MAX_METRIC as described in [TR06] is not sufficient to gracefully shut down links.

We considered the worst-case scenario where the considered link must be shutdown, so that the target metric of the link is MAX_METRIC.

In Figure 6.3, we can see that among the 72 directed links of Geant, the length of the MRMS is 1 for 39 links. In fact, these are the links that can be shut down without causing forwarding loops, so that the reroute sequence only contains MAX_METRIC. Forwarding loops can occur during the shutdown of 33 links. For 30 of them, less then 3 metrics including MAX_METRIC are required. 4 metric changes are necessary for 2 links, and 6 metric changes are necessary for one link. This last link is connecting the Eastern Europe routers to one router in Germany. Eastern Europe routers form a ring, which favours the occurence of forwarding loops, so that many destinations reached via this link have a non-empty Optimized Reroute Metric Sequence.

For the second topology (Figure 6.4), we can see that around 40% of the links require a metric sequences containing more than 2 values. That is, 40% of the links lead to forwarding loops when they are shut down. This confirms the results of our micro-loop analysis performed on the same topology in chapter 4. We also observe that all the obtained reroute metric sequences have a length shorter than 12. 94.1% of them are shorter then 5 and 98.8% shorter than 10. We can see that a small percentage of the reroute metric sequences have a length of 0. These are the sequences for links that are unused in the topology, so that it is not necessary to change their metric before shutting them down.

For the third topology (Figure 6.5), 50% of the links cannot be shutdown directly without causing forwarding loops. This confirms the results of our micro-loop analysis performed on the same topology in chapter 4. Though, 97.3% of the links can be shutdown without forwarding loops by using Reroute Metric Sequences whose length is shorter than 10 and 99.3% with metrics sequences shorter

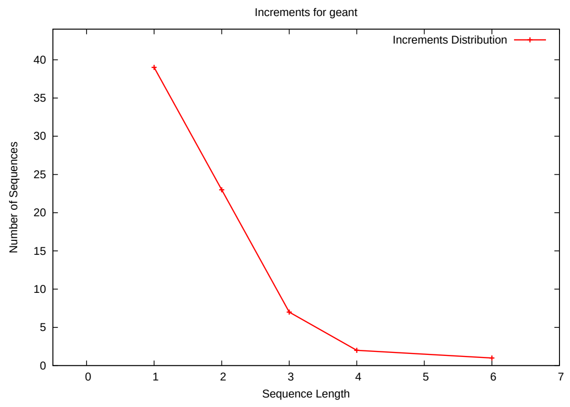


Figure 6.3: Reroute Metric Sequence length distribution for Geant

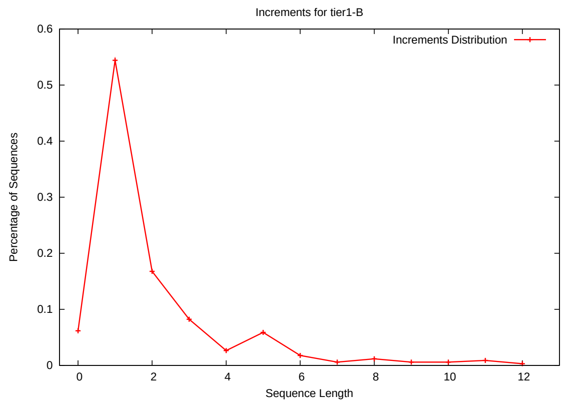


Figure 6.4: Reroute Metric Sequence length distribution for the first Tier-1 ISP

then 20. 5 links require longer metric sequences, with a worst case length of 40 for one link.

Assuming a worst-case convergence time of 5 seconds after a link metric update, applying the solution would let an operator wait for less than a minute to shut down a link without loosing packets in most of the cases. As the solution is applied in the case of planned, non-urgent topological change, the delaying of the actual link shut down seems to be short compared to the obtained gain. When a sudden topological changes occurs while the solution is applied on a link somewhere else in the network, the network monitoring tool should stop the modification of the link metric and restart the computation of a valid Metric Reroute Sequence according to the new topology.

Shutting down a link is a worst-case event for the solution. We also performed analysis where the metric of each link is doubled to consider a case where a metric

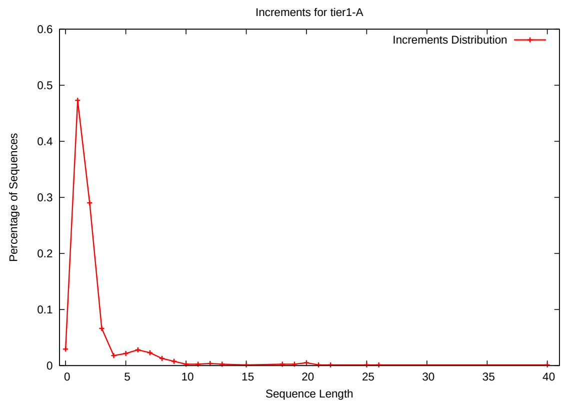


Figure 6.5: Reroute Metric Sequence length distribution for the second Tier-1 ISP

is updated for traffic engineering purposes. For Geant, the maximum length of a sequence was 3. In the second topology, one sequence had a length of 12, and 92% of the sequences where shorter than 3, with the target metric included. In the third topology, the maximum length of a sequence was 22, and the length of most of the remaining sequences was shorter than 5.

6.6 Deployment Issues

The solution described in this paper could be integrated in the network management system (NMS) used by any network operator as we do not require any change to the routers. It could for example be integrated in traffic engineering tools such as NetScope [FGL$^+$00], Cariden MATE [Tec05] or the TOTEM toolbox [BLD$^+$07]. We have implemented the algorithms to compute the MRMS in Java. The TOTEM toolbox can already compute the optimized IGP metrics for a given traffic matrix using [FRT02]. We envision as a further work to provide the MRMS to be used when performing the metric changes that result from the IGP metrics optimization.

Some care must be taken while integrating our solution with an NMS. First, the NMS must maintain the network topology in real-time. The easiest solution is to integrate inside the NMS an IGP monitor such as the one proposed in [SG04a]. With such a monitor, the NMS will always know the exact network topology. Second, it must wait until a metric increment has been applied through the entire network before applying another increment of the sequence. In an IS-IS or OSPF network, timers are configured by the operators in order to control the rate at which routers are allowed to react when a network topology becomes unstable. Some static delaying can be used, or some smarter delaying mechanism can be used to force routers to be reactive when a single link state change occurs, but temper the SPF re-computations and FIB updates when the network becomes unstable, i.e when bursts of link-state changes arise in the topology [Cis04b]. Our method should take this into account when sequenceing the introduction of metric updates. Basically, the delaying between the setting of the link metric should be such that the routers will always consider the event introduced in the network as being a single event, rather than an event being a part of an unstable looking burst of link-state changes. So, when a static SPF_Delay is used, the introduction of the metric updates should be injected by respecting an interval larger then SPF_Delay. When an adaptative Exponential Backoff mechanism is used, the interval between each injection of a metric update must be larger then the $Maximum\_Wait$ interval [Cis04b].

Network operators who care about the convergence inside their network, and who are by such the target users for our technique, configure such timers agressively, which tends to reduce the transition time incurred by our technique. Timers used to control the rate of SPF recomputations in the routers must also be taken into account. Another issue to be considered is that another link can suddenly fail between two metric changes. In this case, the NMS will detect the sudden topology change and recompute the new metric sequence to be applied. Another choice can be to let the network fall back to a best effort convergence process. This would be achieved by directly setting the metric of the link to its target metric.

6.7 Related Work

We already overviewed the work related to loop avoidance in section 5.7. Here, we only specify the position of our scheme relying on metric increments.

The problem of avoiding transient loops during IGP convergence that follows topology changes has mainly been studied by considering extensions to routing protocols. In Chapter 5, we proposed a distributed solution based on messages encoded inside the IS-IS Hello messages exchanged between routers. At the time of this thesis, this scheme was being standardized in [FBS$^+$06], but a few years will pass before the IETF standardizes extensions to OSPF and IS-IS and operators are actually able to deploy them. The main advantage of the solution proposed here is that it can be implemented today in a network management system and does not require any changes to routers and protocols. In [BS07], M. Shand et al. discuss the idea of repeatedly incrementing a link metric by one to reach a forwarding state where the link is no longer used. This solution was rejected by the IETF due to the number of increments that would be required to shut a link down. Our solution only performs the metric updates that are necessary to avoid forwarding loops, so that the idea is now applicable. Shortly after the submission of this thesis, we have been notified by the authors of [IIOY] of this previous work. This paper presents a theoretical analysis of the forwarding loops that can occur for a given destination upon a metric-increase event. The paper also shows that performing the largest loopfree metric increment provides optimal sequences for a given destination. Compared to this prior art, the solution presented in this chapter deals with multiple destinations at the same time, which is required to be practical for the reconfiguration of metrics in IS-IS and OSPF. We also propose techniques to reduce the length of the metric sequences obtained by the merging of the sequences that are loopfree for a single destination, and analyse the applicability of such techniques in ISP topologies.

6.8 Conclusion

In this chapter, we proposed a solution that can be applied now by ISPs to avoid transient forwarding loops during a maintenance operation performed on a link. The solution allows an operator to reconfigure the metric of a link, shut down a link, or set up a link in the network without loosing a single packet. Compared to the solutions proposed before, the main advantage of the solution is that it does not require any modification to the intra-domain routing protocol, as the solution relies on sequences of metric reconfigurations such that each step of the sequence does not disrupt the consistency in the forwarding of packets accross the network. Currently, we do not intend to implement such a solution in the routers themselves, but rather in a network management tool that would issue SNMP or NetConf requests to the node being the head-end of the link to be reconfigured. The provided applicability analysis performed on real ISP topologies shows that the solution never requires a large number of link metric reconfigurations to shut a link down or bring it back up. This is fortunate as the consequence is that applying the solution will not lead to a tremendous delaying of the actual shut down of the link being maintained. Hence, it will not be an important constraint for an operator to use the solution, even if the gain of using it is important. As stringent SLAs are a reality that ISPs currently face, we think that the solution is attractive as it will help them to avoid forwarding loops by themselves while the long lasting standardization process of a protocol built-in solution terminates and implementations reach the market.

Part III

Improving the convergence of inter-domain routing protocols

Chapter 7

IP Fast Reroute for eBGP peering links

We first show by measurements that BGP peering links fail as frequently as intradomain links and usually for short periods of time. We propose a new fast-reroute technique where routers are *prepared* to react quickly to interdomain link failures. For each of its interdomain links, each router precomputes a *protection tunnel*, i.e. an IP tunnel to an alternate nexthop which can reach the same destinations as via the protected link. We propose a BGP-based auto-discovery technique that allows each router to learn the candidate protection tunnels for its links. Each router selects the best protection tunnels for its links and when it detects an interdomain link failure, it immediately encapsulates the packets to send them through the protection tunnel. Our solution is applicable for the links between large transit ISPs and also for the links between multi-homed stub networks and their providers. Furthermore, we show that transient forwarding loops (and thus the corresponding packet losses) can be avoided during the routing convergence that follows the deactivation of a *protection tunnel* in BGP/MPLS VPNs and in IP networks using encapsulation.

7.1 Introduction

To support those mission-critical applications, networks need to guarantee very stringent Service Level Agreements (SLA). When the network is stable and there are no link failures, buffer acceptance, marking and scheduling mechanisms implemented on today's routers [FE05] allow ISPs to provide the performance guarantees required by their customers. Unfortunately, the links used in IP networks are not 100% stable and measurements carried in operational networks indicate that link failures are common events [MIB$^+$04, WJL03, FABK03, GMG$^+$04, FMM$^+$04]. Furthermore, many of those failures only last for a few seconds or tens of seconds.

In the previous part of this thesis, we studied means to improve the recovery of IP networks upon failures of intradomain links. In addition to affecting intradomain links, failures also affect BGP peering links between ASes or links between a BGP/MPLS VPN service provider and a customer site. In this case, ISPs depend on BGP to be able to recover from those failures. Measurements performed recently on high-end routers [Ros04] report an 18 seconds delay to recover the failure of a peering link on a high-end router using 500k BGP routes. Measurements performed in a BGP/MPLS VPN environment [Fil04b] indicate that five seconds is a conservative estimate for the BGP convergence time after the failure of a link between a service provider router and a client site. The current state-of-the-art with BGP routers is thus far from the 50 milliseconds target imposed by stringent real-time applications.

Several authors have proposed modifications to reduce the BGP convergence time in case of failures [PAN$^+$05, GP01]. Those techniques improve the BGP convergence time by reducing the BGP path exploration that must be performed to recover the reachability across the network. However, as they depend on the exchange of messages, the achieved convergence time will always be much larger than the 50 milliseconds target.

Here, we propose a new fast-reroute technique that allows to provide sub-50 milliseconds restoration when a BGP peering link fails, by allowing a local restoration after the failure. We first assume that the failures of the interdomain links are detected by using a trigger from the physical layer such as a SONET loss of signal [VPD04] or a protocol such as BFD [KW06]. This failure detection typically takes less than 15 milliseconds [Fil04b] on high-end routers. Instead of asking routers to *react* to the failure of their BGP peering links by starting an IGP or BGP convergence, our technique *prepares* the routers to quickly handle the failure of such links. For this, each router locates an *alternate nexthop* for each of its BGP peering links. We propose a BGP extension that allows a router to automatically discover the *alternate nexthops* for each of its BGP peering links. When a BGP peering link fails, the router that detects the failure immediately updates its Forwarding Information Base (FIB) to encapsulate the packets that were using the failed link and send them to an *alternate nexthop* through an IP tunnel. The *alternate nexthop* will send the packets to their final destination without using the failed link. On high-end routers, we show how it is possible to modify the FIB within the 50 milliseconds budget. The tunnel to the *alternate nexthop* allows to avoid packet losses, but the packets do not follow the shortest path inside the network. After some time, the router attached to the failed link may need to announce the failure. This will cause a BGP convergence at least inside the local AS. For BGP/MPLS VPNs and IP networks using encapsulation, we show that no packet will be lost in the AS during this convergence. Techniques aimed at reducing BGP path exploration can thus be used in conjunction with our scheme, and benefit from it. Note however that as the reachability is recovered right after the activation of the Fast Reroute technique, a convergence that prevents further packet loss, as described in chapter 8 could be more appropriate.

The remainder of the chapter is organized as follows. In section 7.2, we analyse the failures of BGP peering links in a transit ISP. In section 7.3 we first discuss the problem of protecting interdomain links and show that there are two different problems : the *stub* and the *parallel-links* problems. We then describe the principles of our solution in section 7.4. We show in sections 7.5 and 7.6 how those two problems can be solved by using protection tunnels. Then, in section 7.8 we discuss the conditions under which it is possible to remove an activated protection tunnel without causing packet losses or transient forwarding loops during the routing convergence that follows the deactivation of the protection tunnel. Finally, we compare our proposal with related work in section 7.9.

7.2 Failures of BGP peering links

Several studies have analysed the performance of the global Internet and the impact of link failures from several viewpoints. A first possible way is to collect the link state packets exchanged by routers in a large network and infer the link failures from the reported changes. This method has been applied to several operational ISP networks [MIB$^+$04, WJL03]. Those studies considered different networks, but they basically found three important results. First, *link failures are common events* that must be efficiently handled by the routing protocols. Second, a small number of links are responsible for a large fraction of the failures. This is the common but annoying problem of flapping links. Third, link failures are usually transient events. Very often, the duration of a link failure is around a few or a few tens of seconds.

The second type of study is to use end-to-end measurements or to analyse BGP messages [FABK03, GMG$^+$04, FMM$^+$04] to infer information about link failures. However, it is difficult from such a study to determine the exact location of a failure. To our knowledge, no detailed study has characterised the types of failures that affect eBGP peering links.

7.2.1 Measurement methodology

To evaluate the importance of protecting eBGP peering links, we studied the failures of the eBGP peering links of a transit ISP. In this ISP, the eBGP peering links were configured as follows : a prefix is allocated to each eBGP peering link and the router of the ISP attached to this link advertises this prefix inside its link state packets as long as it considers the link to be up.

When such an eBGP peering link fails, the router attached to the failed link reacts in two steps. First, it advertises a new link state packet without the prefix of the failed peering link. This indicates to all routers of the ISP that the external prefixes advertised via the failed BGP nexthop are now unreachable. All the routers of the ISP will then re-run their BGP decision process to select new routes for the unreachable external prefixes. The second step is that the router will send BGP withdraw messages to indicate that the prefixes learned over the failed eBGP link are not reachable anymore. From an intra-AS routing convergence viewpoint, this exchange of iBGP messages is unnecessary as the failure has already been advertised by the intradomain routing protocol.

To characterise the failures of the eBGP peering links, we first obtained the IP prefixes of all the eBGP nexthops of the studied ISP. We found 47 distinct nexthops. The eBGP sessions with these nexthops were on distinct point-to-point links (SONET/SDH or gigabit Ethernet) as the studied ISP was not attached to Interconnection Points. Thus, the failure the prefix associated to a peering link indicates the failure of this link. All of the peering relationships of the studied AS involved a single peering link with the neighbour AS, except for four neighbour AS's which were each interconnected via two peering links to the studied AS and one neighbour AS which had four peering links to the studied AS.

As the studied ISP is using IS-IS as its intradomain routing protocol, we collected all the IS-IS packets received by a PC running `pyrt`[1] during three months and analysed the collected trace by using `lisis`[2].

7.2.2 Characterisation of eBGP peering failures

We first analysed the IS-IS trace to determine the number of failures of the eBGP peering links. During the studied three-month period, we found 9452 distinct failures. Figure 7.1 provides more details about occurrence of the eBGP peering links failures. The x-axis is the time measured in hours and we list all eBGP peering links on the y-axis so that the failures of the tenth peering link appear on line 10. We use error bars to show both the time of the failure and its duration. However, as most failures are very short, the error bar is often reduced to a simple cross in the figure. Figure 7.1 shows clearly that eBGP failures are regular events and most eBGP sessions are affected by failures[3]. However, the failures were not equally spread among the peering links. In fact, 83% of the failures occurred on a single eBGP peering link. Discussions with the operator revealed that this link had indeed problems at the physical layer that explained the large amount of flapping. Four other links had more than 100 failures during the three month period and some links did not fail at all.

We checked manually the IS-IS trace to determine whether the parallel eBGP peering links with the same neighbour AS failed at the same time. We did not find any common failure among the studied parallel links inside our three-months trace.

The second information that we gathered from the IS-IS trace was the duration of the failures. Figure 7.2 provides the cumulative distribution of the duration of the failures that affected all BGP peering links as well as the most stable eBGP peering links. The curve labelled 'All eBGP peering links' shows than most eBGP peering link failures last less than 100 seconds. However, this number is biased by

[1]`pyrt` is available from `http://ipmon.sprint.com/pyrt`

[2]`lisis` is available from `http://totem.info.ucl.ac.be/tools.html`

[3]As we analysed the prefixes advertised by the routers with IS-IS, a manual reset of an eBGP session is not counted as a failure since it has not effect on IS-IS. The only manual operation that we count as a failure is when the interface is a `shutdown` of a link by the operator.

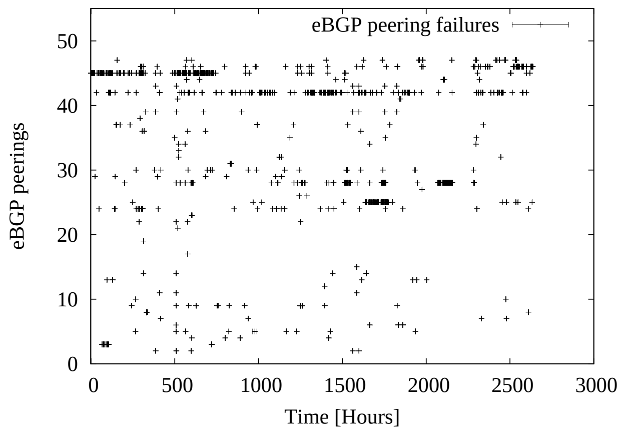


Figure 7.1: Failures of eBGP peerings

the large amount of flapping on some of the studied links.

To reduce this flapping bias, we removed from the analysis the five eBGP peering links that caused most of the failures and draw the curve labelled 'Stable eBGP peering links'. An analysis of the failures affecting the stable BGP peering links reveals several interesting points. First, 22% of the eBGP peering link failures last less than 1 second. Such a transient failure should clearly not cause the exchange of a large number of BGP messages inside the transit AS to converge towards new routes. Second, 82% of the failures of the most stable eBGP peering links lasted less than 180 seconds. This is similar to the study of intradomain link failures reported in [ICM$^+$02],where about 70% of the failures lasted less than 3 minutes. Note that if we consider all eBGP peering links in our analysis instead of only the most stable ones, then 97.5% of the eBGP peering link failures last less than three minutes.

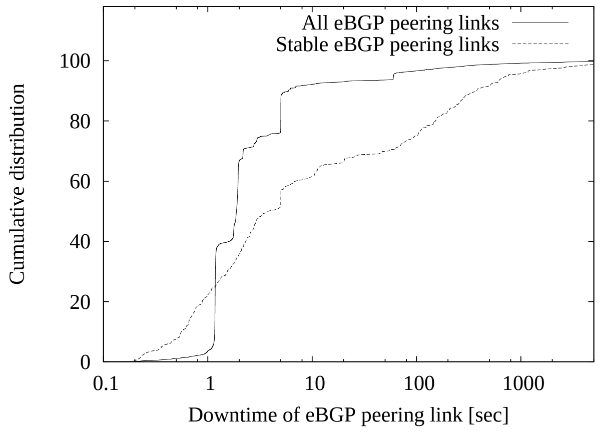


Figure 7.2: Duration of the failures of the eBGP peering links

7.2.3 Implications

Our study confirms the three major results of the intradomain studies. Peering link failures are common events, a small number of peering links are responsible for a large fraction of the failures and peering link failures are usually transient events.

Since most of those failures last less than a few minutes, those events are good candidates to be protected by using a fast reroute technique. Using such a technique and waiting say one minute before advertising the link failure via BGP would help in reducing a lot the BGP churn.

7.3 Problem statement

There are several ways of interconnecting ASes together [WMS04]. To design our fast reroute technique, we first assume that if ASx considers that a BGP peering link with ASy is valuable enough to be protected, then there should at least be a second link between ASx and ASy. This is a very reasonable requirement from an operational viewpoint.

This type of interconnection is very common between transit ISPs and when stub ASes are connected with redundant links to their provider. For such multi-connected ASes, the failure of one interdomain link can be naturally handled by redirecting the packets sent on the protected link to another link with the same AS. For example, in figure 7.3, if link $R1 - X1$ fails, then $R1$ should be able to immediately reroute the packets that were using the failed link to $X2$ via $R2$. This redirection of the packets is possible provided that the same destinations are reachable via the two parallel links. This is a common requirement for peering links [FMR04] and can be a design guideline to provide sub-50 milliseconds recovery in case of failures.

A similar interconnection is also used in BGP/MPLS VPNs (right part of figure 7.3). For important customer sites, it is common to attach two *customer edge* (CE) routers from this site to two different *provider edge* (PE) routers of the service provider. In the right part of figure 7.3, if link $PE1 - CE1$ fails, then $PE1$ should be able to immediately reroute the packets that were using the failed link to $CE2$ via $PE2$.

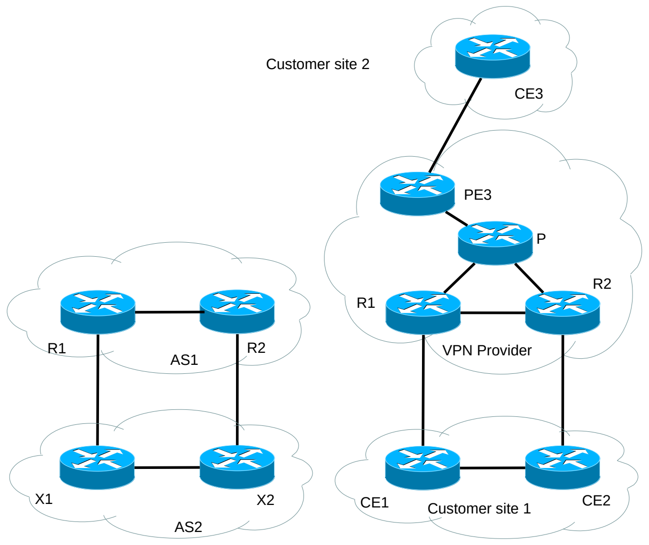


Figure 7.3: The *parallel-links* problem for peering links and BGP/MPLS VPNs

We call the problem of protecting such links the *parallel-links* problem in the remainder of this document. To be deployable, a solution to the *parallel-links* problem will need to meet four requirements.

1. The same solution should be applicable for **both directions** of the interdomain link.

2. As a router controls its outgoing traffic, it should be able to protect it **without any cooperation** with BGP routers outside its AS. This implies that if a tunnel is used, the packet de-encapsulation should be performed in the same AS. A cooperation between routers in neighbouring ASes may improve the performance of the solution, but it should not be required.

3. Links between distinct routers may fail at the same time [VPD04, MIB^{+}04] because they use a shared physical infrastructure (fibre, physical or datalink devices). The set of link that share the same physical infrastructure is usually called a Shared Risk Link Group (SRLG). The solution to the *parallel-links* problem should take into account those SRLGs.

4. The solution should take into account the **BGP policies** [GR00] used for the interdomain links. In most cases when there are multiple links between two ASes, the same BGP policy (e.g. *shared cost peering* or *customer-provider*) is used over all these links. However, the routing policies used between large transit ASes can be more complex. For example, a tier-2 ISP may be a customer of a tier-1 ISP in the US and a peer of the same ISP in Asia. Another example is a corporate network that advertises different prefixes over the multiple links with its provider.

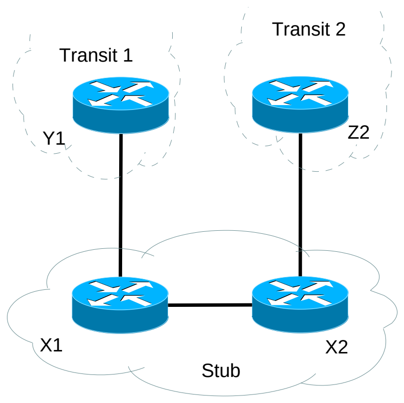


Figure 7.4: The *stub* problem

While requiring the utilisation of parallel-links is reasonable for large ASes, it could be too strong for multi-homed stub ASes. A solution should also be developed to allow a multi-homed stub AS to protect its interdomain links (figure 7.4) when it is attached with a single link to each of its providers. We call this problem the *stub* problem in the remainder of this work. In the *stub* problem, there are two different sub-problems. In the *outgoing stub* problem, the stub AS needs to protect its outgoing packet flow. The solution developed to solve this problem should meet the same requirements as the solution to the *parallel-links* problem as the stub can reach all destinations via either of its two providers.

The second sub-problem is called the *incoming stub* problem. In this case, the stub AS wishes to protect the incoming direction of an interdomain link. The solution developed to solve this problem will require a cooperation between the stub AS and its providers. This cooperation is not a problem as the stub can request the utilisation of a fast recovery technique within the contract with its provider. Furthermore, it should be possible to use the proposed technique to protect one link and not the others. For this, no mutual cooperation between the providers should be required. For example, in figure 7.4 it should be possible for router $Z2$ to protect link $Z2 \rightarrow X2$ without any change to router $Y1$. In figure 7.4, when link $Z2 \rightarrow X2$ fails router $Z2$ should be able to immediately reroute the packets so that they reach the stub without waiting for a BGP convergence.

7.4 Principle of our solution

In this section, we briefly describe the key elements of our proposed solution based on a simple example. Additional details will be provided in the remaining sections. We consider the two transit ISPs shown in figure 7.5 and focus on the packets flowing from the upstream AS to the downstream AS. We assume that the downstream AS advertises the same prefixes over both links and that the routing policies on $X1$ and $X2$ are configured such that $X2 \rightarrow R2$ is used to forward packets while $X1 \rightarrow R1$ is only a backup link. This configuration can be achieved by setting a low `local-pref` value on the BGP routes learned by $X1$.

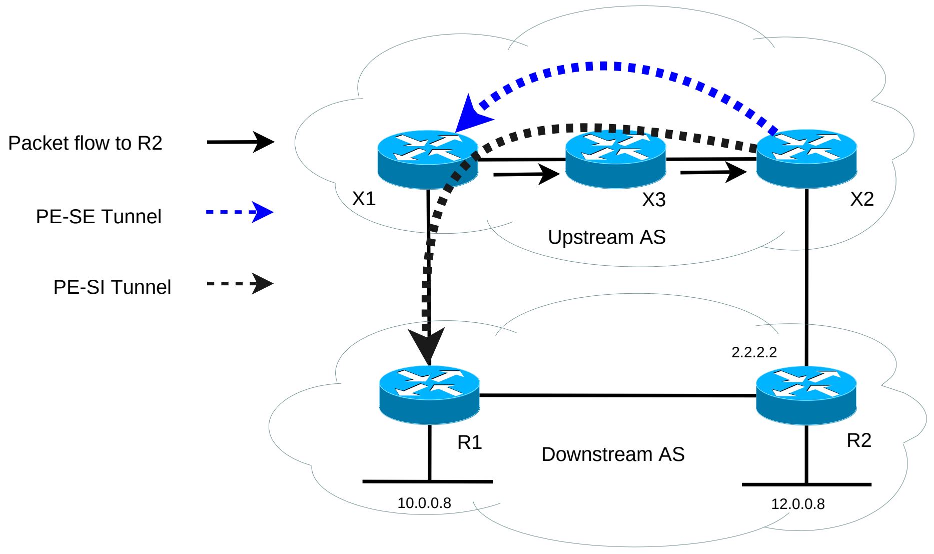
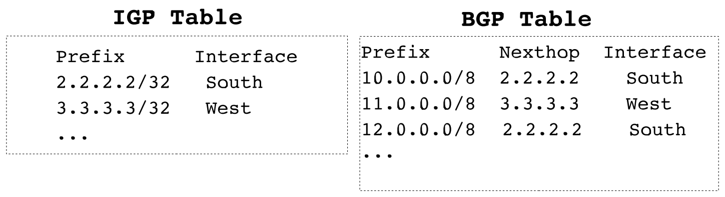


Figure 7.5: Reference network

To quickly react to a failure of directed link $X2 \rightarrow R2$, router $X2$ must be able to quickly update its FIB to send the packets affected by the failure via an alternate path. We describe in section 7.4.1 a technique that allows the FIB to be updated in less than 50 milliseconds. In figure 7.5, the alternate path is clearly through the $X1 \rightarrow R1$ link. Let us assume in this section that router $X2$ was manually configured with this alternate path. We will discuss later mechanisms that allow router $X2$ to automatically discover this alternate path. To forward the packets affected by the failure through the $X1 \rightarrow R1$ link, router $X2$ cannot simply send them on its interface towards $X3$ as $X3$'s BGP table indicates that the nexthop for those prefixes is router $X2$. We show in section 7.4.2 that by using *protection* tunnels it is possible to avoid such loops.

7.4.1 A fast update of the FIB

The update of the FIB after the failure is a key implementation issue to achieve the sub-50 milliseconds target. The FIB is a data structure that associates a BGP prefix to a nexthop and an outgoing interface. Figure 7.6 shows the conceptual view of such a FIB as two tables. In such a FIB, the outgoing interface is obtained from the IGP routing table. Detailed measurements performed on high-end routers revealed that the time required to update one entry of such a FIB was on average around 110 microseconds per entry [FFEB05]. This implies that less than 5000 FIB entries can be updated within the sub-50 milliseconds target on such routers.



IGP Table

Prefix	Interface
2.2.2.2/32	South
3.3.3.3/32	West
...	

BGP Table

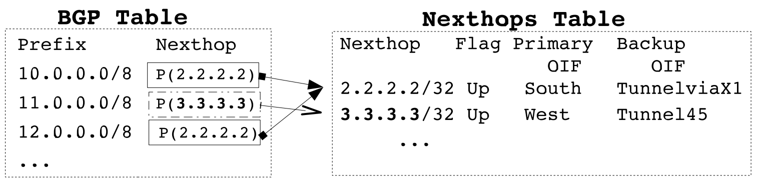
Prefix	Nexthop	Interface
10.0.0.0/8	2.2.2.2	South
11.0.0.0/8	3.3.3.3	West
12.0.0.0/8	2.2.2.2	South
...		

Figure 7.6: Classical conceptual organization of the FIB

To achieve the sub-50 milliseconds target it is necessary to reduce the number of FIB entries that must be modified after the detection of a failure. There are several possible methods to reroute packets towards many destinations without changing a large number of entries in the FIB. Some commercial routers already support such mechanisms [Cis04a]. The exact organization of the FIB strongly depends on the hardware capabilities of the concerned router. The details of those FIB organizations are outside the scope of this work. We show, conceptually, one possible organization of the FIB to illustrate the possibility of achieving this fast reroute.

This new organization of the FIB is illustrated in figure 7.7. Conceptually, this FIB is organized as two tables. The first table contains the BGP prefixes and the BGP nexthops are *pointers* to a table (noted *P(...)*) of all nexthop entries. Each nexthop entry in the second table contains the address of the nexthop, a flag that indicates whether the link to the nexthop is up or down and two outgoing interfaces (OIF) : a primary OIF and a secondary OIF. The OIF is in fact a data structure that contains all the information required to forward packets on this interface. For a point-to-point interface, this data structure will contain the layer 2 encapsulation to be used (e.g. PPP or Packet over SONET). For a point-to-multipoint interfaces, the data structure will contain the layer 2 encapsulation and the layer 2 address of the nexthop router. For a virtual interface such as a tunnel, the FIB will contain the IP address of the tunnel endpoint and the tunnel specific parameters. Those parameters are useful notably for L2TP [LTG04] or MPLS over IP tunnels [WRR04].

With this new FIB, when the router consults its nexthop table, it uses the primary OIF if the flag is set to “Up” and the backup OIF otherwise. This means that when a peering link fails, a *single* modification to the Nexthops Table is sufficient to reroute *all* affected prefixes over a *protection* tunnel. This clearly meets the sub-50 milliseconds target.



Prefix	Nexthop
10.0.0.0/8	P(2.2.2.2)
11.0.0.0/8	P(**3.3.3.3**)
12.0.0.0/8	P(2.2.2.2)
...	

Nexthop	Flag	Primary OIF	Backup OIF
2.2.2.2/32	Up	South	TunnelviaX1
3.3.3.3/32	Up	West	Tunnel45
...			

Figure 7.7: Improved conceptual organization of the FIB

7.4.2 The protection tunnels

As explained earlier, a solution is required to allow router $X2$ to reroute the packets immediately to router $X1$ even if the routing tables of $X3$ and $X1$ still point to $X2$ as their nexthop. For this, two different types of tunnels can be envisaged :

- A tunnel from the *primary egress* router (X2) to another egress router (e.g. X1) of the upstream AS that peers with the same downstream AS. We call this tunnel a *primary egress - secondary egress* or *pe-se* tunnel.

- A tunnel from the *primary egress* router (X2) to another ingress router in the downstream AS (e.g. R1). We call this tunnel a *primary egress - secondary ingress* or *pe-si* tunnel.

The *pe-se* and *pe-si* protection tunnels are “pre-defined” before the link failure. At the *primary egress* router, a protection tunnel is defined by two parameters : an encapsulation header and an outgoing interface. At the secondary *ingress* or *egress*, the definition of the protection tunnel is simply the ability to de-encapsulate the packets received over the tunnel.

Several types of protection tunnels exist : IP over IP, GRE, IPSec, L2TP, MPLS over IP, ... However, not all encapsulation types are suitable for *pe-se* tunnels. Consider again figure 7.5. When link $X2 \rightarrow R2$ fails, router $X2$ will encapsulate the packets towards router $X1$. If $X2$ uses IP-in-IP encapsulation, then router $X1$ will use its FIB to forward the de-encapsulated packets. Unfortunately, $X1$’s FIB may still use $X2$ as the nexthop to reach the affected prefixes.

To avoid this problem, we require the utilisation of an encapsulation scheme that contains a label such as L2TP [LTG04] or MPLS over IP [WRR04]. This label is assigned by the *secondary egress* router. When it receives an encapsulated packet, it uses the label as a key to forward the de-encapsulated packet over the appropriate secondary link *without consulting its BGP FIB*. This ensures that the *secondary egress* will not return the received encapsulated packets to the *primary egress* router even if this *primary egress* is the current BGP nexthop according to the FIB of the secondary egress router.

Using IP-based tunnels usually raises two immediate questions. The first one is the cost of encapsulation and de-encapsulation. In the past, those operations were performed on the central CPU of the router and were costly from a performance viewpoint [Rek91]. Today, the situation is completely different and high-end routers are able to perform encapsulation or de-encapsulation at line rate. Furthermore, many large ISPs have deployed MPLS to support BGP/MPLS VPNs and some rely on L2TP or GRE-based encapsulation [Gro05]. The second question is the problem of fragmenting packets whose size exceeds the MTU. On current Packet over SONET interfaces used by high-end routers, this issue becomes a design problem : the network must be designed to ensure that the MTU is large enough. The design guidelines developed for GRE-based tunnels in [Gro05] would ensure that fragmentation is avoided when IP-based protection tunnels are used.

In a production network, allowing routers to process encapsulated packets may cause security problems unless the routers have a way to verify that the packets come from legitimate sources. For the *pe-se* tunnels, the tunnel source belongs to the same ISP as the tunnel destination. In this case, IP-based filters such as those already deployed by ISPs [GS02] would be sufficient. For the *pe-si* tunnels, the *secondary ingress* should be able to verify the validity of the received encapsulated packets. A possible solution could be to use IPSec for those tunnels. Another solution would be to use filters.

To define a *pe-se* (resp. *pe-si*) protection tunnel, the *primary egress* router must thus determine the IP address of the appropriate *secondary egress* (resp. *secondary ingress*) router and the tunnel type to be used. We propose in the following sections techniques to select the endpoints of the protection tunnels.

7.5 The *parallel-links* problem

To solve the *parallel-links* problem, we utilise *pe-se* protection tunnels. Such tunnels could be configured manually on the *primary-egress* router. For example, the network operator could configure on this router the addresses of the candidate *secondary-egress* routers and the parameters of the *pe-se* tunnel to be used. This manual configuration would be sufficient in the common case where a small stub AS is connected to its provider via two interdomain links. However, in a large network, an auto-discovery mechanism is required to simplify the configuration and more importantly to allow the routers to automatically adapt the protection tunnels to topology changes.

To build this auto-discovery mechanism, we first consider the simple case of two physically independent parallel links and assume that the same prefixes are advertised by the downstream AS over those links. In this case, the main problem for the *primary egress* router is to locate the appropriate *secondary egress* router. To discover the *secondary egress* router, the *primary egress* router cannot simply cannot simply consult its BGP table as it may not have alternate routes for the affected prefixes. For example, in figure 7.5, router $X2$ does not learn any route advertised by the downstream AS from router $X1$ due to the `local-pref` settings on this router. A similar situation could occur in a large AS, where due to the utilisation of BGP confederations or route reflectors, routers only receive a single route towards each destination.

To solve this auto-discovery problem, we propose to allow each egress router to advertise via iBGP the "characteristics" of its currently active eBGP sessions by using a new type of BGP routes called *protection routes*. A *protection route* contains the following information :

- the NLRI is the local IP address on the peering link with the downstream AS.
- the AS-Path attribute contains only the downstream AS
- a tunnel attribute containing the parameters of the protection tunnel to be established

The IP address used in the NLRI must be routable and unique, at least within the upstream AS. The uniqueness of the NLRI information is necessary to ensure that the *protection route* will be distributed to all the routers inside the upstream AS. If the same NLRI was used for several protection routes, then a route reflector could run the BGP decision process to advertise only one of them to its clients. By using a unique NLRI for each protection route, we ensure that the protection route is distributed throughout the AS even if there are route reflectors or confederations. The tunnel attribute indicates the supported type of tunnel (GRE , L2TP or MPLS over IP tunnels) and the optional parameters such as the label for MPLS over IP encapsulation.

It is important to note that a router advertises *one* protection route for each of its active eBGP sessions. A *protection route* is only advertised when the corresponding BGP peering link is active. When a peering link fails, the corresponding *protection route* is withdrawn. Furthermore, the protection routes are only distributed inside the local AS. For these reasons, the iBGP load due to the protection routes is negligible compared to the normal iBGP load.

When the *primary egress* router needs to select a *pe-se* tunnel endpoint for a primary link, it considers as candidate *secondary egress* routers all the protection routes whose AS-Path is equal to the downstream AS and whose tunnel endpoint is reachable according to its IGP routing table. In practice, the closest *secondary egress* would often be the best one.

However, as discussed in section 7.3, the solution should also be able to protect from SRLG failures. To be able to correctly handle SRLG failures, the routers need to know the SRLG associated with each BGP peering link. For example, considering figure 7.8, router $R2$ should not be selected as a *secondary egress* to protect link $R1 \rightarrow X1$ as link $R2 \rightarrow X1$ also terminates at router $X1$. In practice, a BGP peering link can be characterised by a set of SRLG values specified by the network operator [VPD04]. A BGP peering link is composed of two half-links, one half in the upstream AS and the other in the downstream AS. It will thus be characterised by SRLG values managed by the downstream AS and SRLG values managed by the upstream AS. The SRLG values can be manually configured on a per eBGP session basis by encoding each value as a pair `AS#:SRLG-value` of 32 bits integers[4] where `AS#` is the AS number of the AS that allocated the SRLG value.

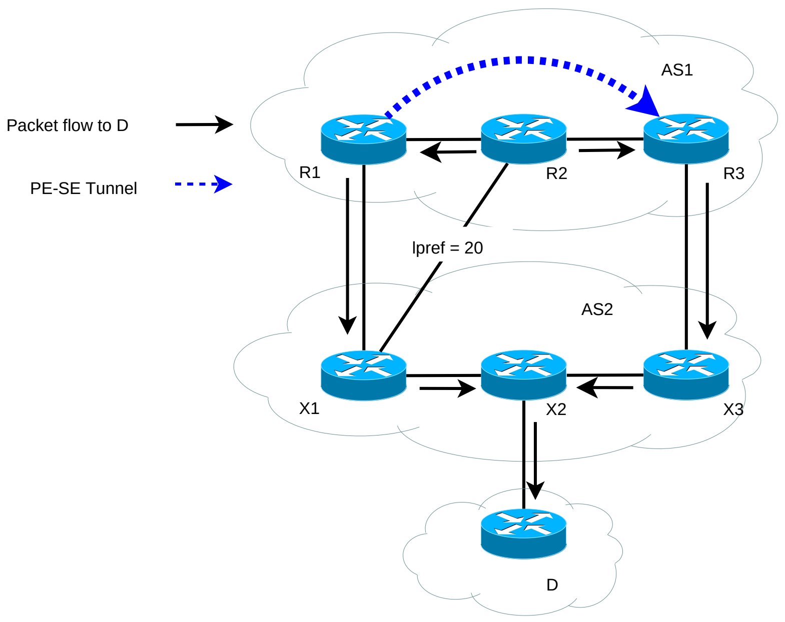


Figure 7.8: Utilisation of a *pe-se* protection tunnel

Another problem to be considered is when different BGP policies are used over the parallel-links. As an example, consider the network topology shown in figure 7.8. Assume that *primary egress* router `R1` needs to create a protection tunnel for directed link `R1`→`X1` and that `R1` and `R3` receive a full routing table while `R2` only receives the client routes of AS2. In this case, router `R1` should select `R3` as its *secondary egress* since `R3` receives the same routes as `R1`.

To solve this problem, each egress router must know the BGP policy used by its peer. This is because the packets that are sent on the *primary-egress* → *primary ingress* link depend on the BGP routes advertised by the *primary ingress* router. For this, we propose to add to the configuration of each eBGP session an identifier of the BGP policy used (customer, peer, . . .). In practice, this identifier would

[4]The Traffic Engineering extensions to OSPF and IS-IS already encode SRLG values as 32 bits integers.

Parameter	Comment
NLRI	IP address of the egress router on the peering link
AS-Path	Downstream AS
eBGP session type	32 bits unsigned integer
Tunnel attribute	Type and optional parameters for the tunnel
SRLG	optional list of pairs `AS#:SRLG-value`
Link bandwidth	optional extended community

Table 7.1: Proposed protection routes

usually correspond to the peer-group used to specify the export filter [WMS04]. Each egress router should thus be configured with the BGP policy used by its peer. To reduce the amount of manual configuration, the eBGP session type could be exchanged during the establishment of the BGP session by encoding this information inside the BGP capabilities. If required, BGP capabilities can also be updated during the lifetime of the BGP session. The SRLG values could be exchanged over the eBGP session by using the same technique.

Coming back to the example of figure 7.8, `R3` will advertise a protection route for an eBGP session of type 0 and `R2` a protection route for an eBGP session of type 1. `R1` will select the protection route of type 0 and `R3` will be the endpoint of the *pe-se* protection tunnel.

Finally, parallel links between ASes can have different bandwidth. When the endpoint of a protection tunnel is chosen, it should be possible to select as tunnel endpoint a secondary egress router with sufficient capacity. For this, the protection route can optionally contain the bandwidth extended community defined in [STR04]. Table 7.1 summarises the content of protection routes.

When the *primary egress* router needs to select a *pe-se* tunnel endpoint to protect a primary link, it will consider all the protection routes whose AS-Path contains the downstream AS and whose tunnel endpoint is reachable according to its IGP routing table. The selection of the best protection route among those candidates will be done as follows.

1. Remove from consideration the protection routes with an eBGP session type which differs from the eBGP session type of the primary eBGP session.

2. Remove from consideration the protection routes that contain one of the SRLG values associated to the primary link to be protected.

3. If there are still several candidate protection routes, break the ties by using the IGP cost to reach the tunnel endpoint and, if available, the link bandwidth extended community.

If there is congestion inside the upstream AS, it is also possible to utilise traffic engineered *pe-se* tunnels. A traffic engineered MPLS tunnel with bandwidth reservations can be established by the *primary egress* to reach the *secondary egress* by using RSVP-TE. This type of tunnel ensures that sufficient bandwidth will be available for the protected traffic, but of course it forces the routers to maintain additional state.

7.6 The *stub* problems

To solve the stub problems, we have to consider the two directions of the packet flow. For the *outgoing stub* problem, we note that in this case the stub receives either a default route or a full BGP routing table from its providers. Thus, the same destinations are reachable over all links with the providers. For this reason, we propose the utilization of a *pe-se* protection tunnel to solve the *outgoing stub* problem. For the *incoming stub* problem, we will utilize a *pe-si* protection tunnel.

7.6.1 The *outgoing stub* problem

To protect the *stub→provider* packet flow on an interdomain link, we note that from the stub's viewpoint, the providers can be considered as equivalent as they can be used to reach any destination. Thus, the *outgoing stub* problem is similar to the *parallel links* problem. We simply propose to reserve the value 0 for the eBGP session type corresponding to an eBGP session over which a full BGP routing table is advertized and slightly change the criteria to select the best *secondary egress* router for the protection tunnel. When the eBGP session type of the primary link is equal to 0, the selection is done by considering all the protection routes with an eBGP session type of 0 independently of their AS-Path. The selection of the best protection route among those candidates is done as follows :

1. If the type of the eBGP session of the primary link is 0, remove from consideration the protection routes with a strictly positive eBGP session type. If the type of the eBGP session on the primary link is strictly positive, remove from consideration the protection routes with an eBGP session type which is strictly positive and differs from the eBGP session type of the primary eBGP session.

2. Remove from consideration the protection routes that contain one of the SRLG values of the primary link to be protected.

3. If protection routes whose AS-Path is equal to the downstream AS exists, remove all the other protection routes. This rule is not mandatory. Its application conceals the protection via the same AS as the initial one.

4. Finally, select protection routes on the basis of the IGP cost to reach the tunnel endpoint and, if available, the link bandwidth extended community.

For example, consider in figure 7.9 that `AS1` is a stub and that `P1`, `P2` and `P3` are its providers. Assume that `P2` and `P1` advertise a default route and `P3` only regional routes. In this case, `R2` will advertise inside `AS1` two protection routes :

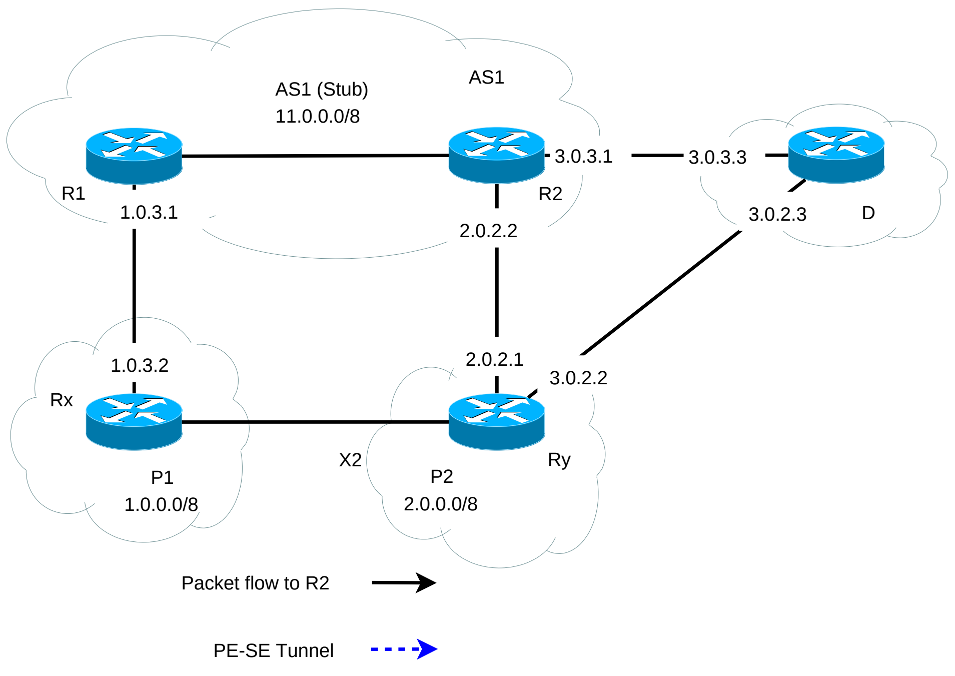


Figure 7.9: A stub AS attached to three providers

- a protection route with `NLRI=2.0.2.2`, `AS Path=P2`, and `eBGP session type=0`
- a protection route with `NLRI=3.0.3.1`, `AS Path=P3`, and `eBGP session type=17`

To protect link `R1`→`RX`, `R1` would select IP address `2.0.2.2` as the endpoint of the protection tunnel.

7.6.2 The *incoming stub* problem

To quickly recover the *provider*→*stub* packet flow when an interdomain link to a stub fails, we propose to rely on a *pe*-*si* protection tunnel. This tunnel is established between the *primary egress* router located inside one provider and a *secondary ingress* router inside the stub. The advantage of using a *pe*-*si* tunnel in this case is that the routers of the secondary provider are not involved neither in the activation of the protection tunnel nor in the de-encapsulation of the packets.

As for the *pe*-*se* protection tunnel, the best *secondary ingress* router and the parameters of the protection tunnel to be used can be manually configured on the *primary egress* router. This manual configuration is probably acceptable for a small dual-homed stub AS, but it increases the complexity of the configuration that must be maintained by the operators. A better solution is to use BGP to auto-configure the required *pe*-*si* protection tunnels.

For this, we propose to allow each ingress router in the stub AS to advertize over the eBGP session with its provider the *secondary ingress* routers inside the stub that could be used as candidate endpoints for *pe-si* protection tunnels. This information can be advertized by the *primary ingress* router as *protection routes*[5]. In those protection routes, the NLRI is set to the IP address of the *secondary ingress* router and the tunnel attribute contains the supported tunnel type and the associated tunnel parameters.

A key issue for the utilization of a *pe-si* protection tunnel is that the *primary egress* router must still be able to reach the *secondary ingress* router even if it was using the failed link to the *primary ingress* router to reach all the IP prefixes advertised by the stub. This reachability can be guaranteed provided that the IP address of the *secondary ingress* router belongs to an IP prefix allocated to and advertized by the secondary provider and not to an IP prefix advertized by the stub. This is a common practice among ISPs and could become a design rule when *pe-si* tunnels are required. For example, in figure 7.9, router `RX` learns prefix `11.0.0.0/8` from router `R1`. If link `RX` → `R1` fails, router `RX` can still reach the *secondary egress*, `R2`, by sending encapsulated packets to IP addresses `2.0.2.2` or `3.0.0.3.1`.

The *protection routes* that are advertised by the *primary ingress* router can be manually configured, but a better solution is to use the protection routes that are distributed inside the stub to solve the *outgoing stub* problem.

For this, each ingress router of the stub AS will filter the *protection routes* that it receives via iBGP. The ingress router will only advertise over its eBGP session the protection routes containing the same eBGP session type as the session type of the primary link and different SRLG values than the SRLG values associated to the primary link.

The *primary egress* router will select, among the protection routes that it receives over its eBGP session, the best endpoint for the *pe-si* protection tunnel.

For example, consider the stub `AS1` attached to providers `P1`, `P2` and `P3` in figure 7.9. Assume now that the three providers advertise a default route to the `AS1`. `R1` will receive via iBGP two protection routes from router `R2` :

- a protection route with `NLRI=2.0.2.2`, `AS Path=P2`, and `eBGP session type=0`
- a protection route with `NLRI=3.0.3.1`, `AS Path=P3`, and `eBGP session type=0`

On its eBGP session with `RX`, `R1` will advertise these two *protection routes* with `3.0.3.1` and `2.0.2.2` as tunnel endpoints. Based on the received candidate protection routes, `RX` will select `2.0.2.2` as the tunnel endpoint to protect the `RX`→`R1` link.

[5]The `NO_ADVERTISE` BGP community is attached to the protection routes advertised over eBGP sessions as they do not need to be distributed beyond the *primary egress* router.

7.7 Non covered cases

Our solution does not automatically cover the specific cases where a transit AS is connected with a single link to another transit AS. Indeed, in such a scenario, there is no simple way for a router to find out a tunnel endpoint, thus via another transit AS, that provides the same (or more complete) reachability as the tail-end of protected peering link. However, we argue that such cases are very rare as transit ASes that are concerned about the resiliency of their networks tend to establish redundant peering links with the relevant transit ASes to which they are connected. Also, in the BGP MPLS/VPN case, which is the most important case for many ISPs, these cases do not apply, as a VPN site that is connected to its provider with a single link is disconnected from the VPN upon its failure.

7.8 BGP convergence after deactivation of a protection tunnel

Once activated, a protection tunnel helps in forwarding the packets that would have initially been forwarded along the failed link over an alternate path. However, when a protection tunnel is used, the packet flows inside the network do not follow the paths that they should follow when the corresponding peering link is removed. In other words, the application of the protection tunnel is not equivalent to a regular BGP convergence. If the failure lasts for a few seconds, this is not a problem, but using a protection tunnel for several minutes or hours is not recommended, as forwarding paths do not match the paths advertised in BGP.

The measurements discussed in section 7.2 have shown that most of the failures of eBGP peering links are short. A primary egress that activated a protection tunnel should thus wait some time before advertising the failure of its peering link via BGP or its IGP. If the failure is short enough, the peering link will come back while the protection tunnel is still active. At that time, the primary egress router simply needs to modify its FIB to deactivate the protection tunnel. Otherwise, the advertisement of the failure will trigger the exchange of iBGP messages and the update of the FIBs of many routers, although the network will be brought back in its initial state soon, hence triggering another wave of iBGP messages and FIB updates.

If the failure lasts long, then the failure should be reported to BGP and the network should converge. To meet the requirements expressed in section 7.3, we must ensure that no packet will be lost during this BGP convergence.

To illustrate the potential problems caused by the iBGP convergence, let us consider the network topology shown in figure 7.10 and focus on the packets sent to destination *D*. In this topology, `R1-X1` is the primary link between AS1 and AS2 and `R3-X3` a backup link. This backup link is implemented by configuring a low `local-pref` attribute in the import filter of router `R3`. When link `R1-X1` fails, the *pe-se* tunnel reroutes the packet via link `R3-X3`. However, the utilisation

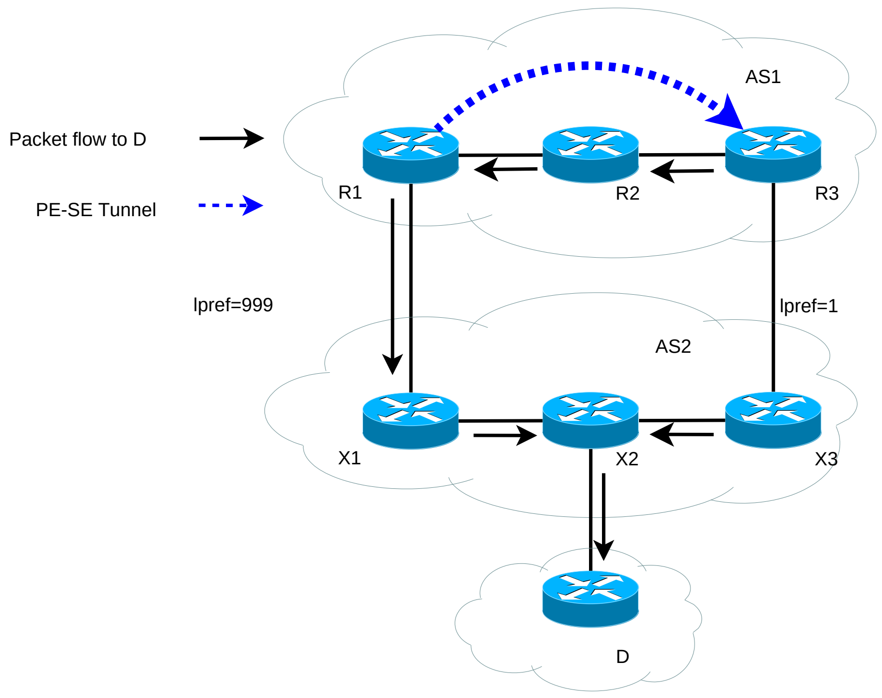


Figure 7.10: Example topology for the deactivation of a *pe*-*se* tunnel

of this tunnel is not optimal since the packets that enter `AS1` at router `R2` will pass twice through the `R1-R2` link. After some time, router `R1` will need to remove the *pe*-*se* protection tunnel. If router `R1` sends a BGP withdraw message (W_{R1}) to indicate that destination *D* is not reachable anymore, router `R3` will react to this withdraw message by updating its FIB and sending a BGP update indicating its own route (U_{R3}). Depending on the processing order of those messages by the routers, several transient losses of connectivity to destination D are possible. In table 7.2, we use the notation $Rx : W_{R1}$ (resp. $Ry : U_{R3}$) to indicate that message W_{R1} (resp. U_{R3}) has been processed by router Rx (resp. Ry). As shown by this table, only one ordering of the updates of the FIBs ensures the reachability of D during the convergence. For five of the possible orderings, D becomes unreachable during a short period of time and a transient loop between $R1$ and $R2$ appears for two of the possible orderings. Note that the transient loops only occur in the context of an AS that does not use encapsulation to forward packets across its network.

Thus, two different problems must be solved to allow a *pe* router to remove a *pe*-*se* protection tunnel without causing packets losses :

- All the destinations that are currently reached via the protection tunnel must remain reachable during the entire routing convergence (*the convergence preserves reachability*)

First BGP message	Second BGP message	Third BGP message	Fourth BGP message	Comment
$R2 : W_{R1}$	$R3 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D unreachable from R2 between first and third message
$R2 : W_{R1}$	$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	D unreachable from R2 between first and fourth message
$R3 : W_{R1}$	$R2 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D unreachable from R2 between second and third message
$R3 : W_{R1}$	$R2 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	D unreachable from R2 between second and fourth message
$R3 : W_{R1}$	$R2 : U_{R3}$	$R2 : W_{R1}$	$R1 : U_{R3}$	D always reachable during convergence
$R3 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	$R2 : W_{R1}$	transient loop R1-R2 between third and fourth message
$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : W_{R1}$	$R2 : U_{R3}$	D unreachable from R2 between third and fourth message
$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	$R2 : W_{R1}$	transient loop R1-R2 between second and fourth message

Table 7.2: Processing order of the iBGP messages inside AS1 after the transmission of a BGP withdraw

- No transient packet forwarding loops are caused by the update of the FIBs of the routers inside the AS (*the convergence does not cause transient loops*)

7.8.1 Forwarding schemes

To preserve reachability and avoid transient loops, we need to consider how packets are forwarded inside an autonomous system. This problem was discussed early during the development of BGP [Rek91] and two techniques have emerged. The first solution, proposed in 1990, is to use encapsulation [HKM$^+$90], i.e. the ingress border router encapsulates the interdomain packets inside a tunnel towards the egress border router chosen by its BGP decision process. At that time, encapsulation suffered from a major performance drawback given the difficulty of performing encapsulation on the available routers [Rek91]. Today, high-end routers are capable of performing encapsulation and decapsulation at line rate when using MPLS or IP-based tunnels [Gro05].

The second technique, called *Pervasive BGP* by [RG94] is to use BGP on all (border and non-border) routers inside the transit autonomous system. This technique is still used in pure IP-based transit networks. We explain in section 8.5 the difficulty of avoiding transient forwarding loops during a convergence inside an autonomous systems using *Pervasive BGP*. Roughly, we will explain that if a loopfree convergence is feasible, any possible solution does not scale well and would thus be impractical.

7.8.2 Deactivation of protection tunnels for BGP/MPLS VPN peering links

In a network providing BGP/MPLS VPNs (figure 7.11), iBGP is used to distribute the VPN routes to the PE routers [RR99]. A VPN route is composed of two parts : a *Route Distinguisher (RD)* and an IP prefix. The *RD* is used to allow sites belonging to different customers to use the same IP addresses (e.g. RFC1918 private addresses). A VPN route is considered as an opaque bit string by the BGP routers that distribute the routes. A service provider can either use the same *RD* for all VPN routes belonging to the same VPN or a different *RD* for each PE-CE link. Furthermore, a *route target (RT)* is associated to each VPN route. A *RT* is encoded as a BGP extended community. It is used, in combination with filters on the PE routers, to ensure that a VPN route from a given customer is only distributed to the PE routers that are attached to CE routers belonging to the same VPN. This utilisation of the *RT* reduces the size of the VPN routing tables on the PE routers [RR99].

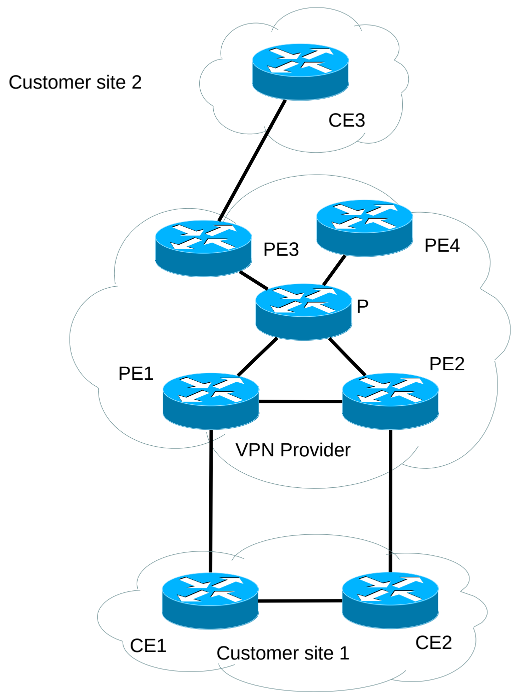


Figure 7.11: Example with BGP/MPLS VPNs

To avoid packet losses during the BGP convergence in this context, the service provider simply needs to configure its PE routers to use a different *RD* for each PE-CE link. Using a different *RD* ensures that each PE router will receive via iBGP all the VPN routes for the prefixes that are reachable over the PE-CE links.

This remains true even if the service provider network is divided in confederations or uses BGP route reflectors as VPN routes with different *RD* are considered as different *opaque* prefixes by the BGP decision process. When a `PE` router sends BGP withdraw messages due to the failure of a parallel-link, those messages will reach distant `PE` routers where an alternate VPN route (with a different *RD*) is already available. As this alternate route uses an MPLS tunnel, it is loop-free. The same reasoning applies if the service provider uses IP tunnels instead of MPLS tunnels.

For example, consider in figure 7.11 the failure of link `PE1-CE1`. `PE1` first activates the *pe-se* protection tunnel to reach `CE2` via `PE2`. At that time, `PE3` uses an MPLS tunnel to send via `PE1` the VPN packets from `CE3` to `CE1`. Then, `PE1` sends a BGP withdraw message. When this message reaches `PE3`, it updates its VPN routing table and uses the loop-free MPLS tunnel to `PE2` to reach `CE2` and `CE1`.

If using distinct route distinguishers is not part of the AS policy, the solutions proposed for regular, non VPN, BGP routes can be applied.

7.8.3 Deactivation of protection tunnels for regular peering links

Deactivation of a *pe-se* tunnel

In chapter 8, we consider the problem of shutting down a BGP peering link for maintenance purposes. Once a *pe-se* protection tunnel is activated, a fast convergence of BGP should not be performed at the cost of packet loss, as the reachability of affected destinations is still ensured through the network. Our fast reroute technique thus turns a sudden peering link failure into a non urgent one, so that the make-before-break convergence proposed in section 8.4 is fully applicable. Note that we recommend to use the fully automatic shutdown solution to accomplish this task as a reconfiguration of routers to deactivate a protection tunnel would not be practical.

Deactivation of a *pe-si* protection tunnel

When a *pe-si* protection tunnel is used, it is possible that no alternate paths are available at the borders of the AS, so that an interdomain graceful shutdown mechanism, as described in section 8.4.4, might be required.

7.9 Related work

Several fast reroute techniques have been proposed and are deployed in MPLS networks. A survey of these techniques may be found in [VPD04]. Several ISPs have started to deploy interdomain MPLS tunnels. Extensions to RSVP-TE to allow those tunnels to be protected on the interdomain links have been proposed recently [CP04]. The main advantage of our solution is that it allows to quickly recover from the failure of PE-CE links in BGP/MPLS VPNs although no MPLS tunnel is used on those links.

To our knowledge, our solution is the first pure IP Fast Reroute technique that allows to protect interdomain links. In [Rei04] an extension of the O2 routing protocol [SCK$^+$03] was proposed to recover from the failure of interdomain links. However, this solution assumes both a new routing protocol and that the *primary* and *secondary egress* routers are directly connected.

Gummadi et al. propose in [GMG$^+$04] a source routing technique that allows endsystems to reroute packets around failures by using intermediate nodes as relays. Measurements with a prototype implementation reveal that this technique allows to recover from 56% of network failures. This end-to-end recovery technique is characterised by a recovery time of at least several seconds. Our fast-reroute mechanism only allows to recover from a failed BGP link, but those links are key in today's Internet. Our technique is also applicable for the BGP/MPLS VPNs that are increasingly used to replace frame relay and ATM-based networks.

Several modifications to BGP have been proposed to reduce the BGP convergence time. To our knowledge, the closest solution to our interdomain tunnels is the Fast Scoped Rerouting proposed for BGP in [BLSZ03]. With this approach, BGP routers try to find an alternate path for each destination affected by a failure and exchange messages with the routers on this alternate path. As BGP messages must be exchanged *after* the failure to find an alternate path, the recovery time of this BGP extension will be longer than with our solution. The Root Cause Notification proposed in [PAN$^+$05] adds to the BGP messages an information about the reason for the BGP message. Another method to tag BGP messages was proposed in [CDZK05]. This solution is much similar to RCN but assumes multiple routers per AS. Our solution is orthogonal to those BGP extensions and could benefit from them if implemented and deployed. However, as our solution allows the protection tunnel to recover the reachability across the network directly after the failure without requiring the exchange of any BGP message, the BGP convergence following the activation of the protection tunnel should be performed by avoiding further losses of packets. This could be achieved by using the solutions proposed in chapter 8.

A next step to this work has been proposed in [Fil06]. Roughly, this improvement of the solution relies on a per-prefix backup entry stored in the FIB, and a tight binding of the IGP FIB with the state of the BGP FIB. Such a technique greatly improves the flexibility of the solution, and allows a fast adaptation of the FIB of the routers upstream to the failure location, which reduces the overhead incurred by the local protection mechanism.

7.10 Conclusion

BGP peering links are important in both the global Internet and in BGP/MPLS VPNs. We have analyzed the stability of eBGP peering links in a transit AS and have shown that those links often fail, usually for short periods of time.

In this work, we have proposed a new technique to ensure that the packet flow on failed eBGP peering links can be recovered within 50 milliseconds. Our solution relies on two types of *protection tunnels*. Its main advantages are that it can be incrementally deployed, does not require major changes to the BGP protocol and is applicable for both normal BGP peering links and for the links to customer sites in BGP/MPLS VPNs.

The *primary egress-secondary egress* protection tunnels can be used when there are several parallel links between two ASes. We have proposed simple BGP extensions that allow border routers to automatically discover the best *pe-se* protection tunnel to use to protect each of their interdomain links. In autonomous systems using encapsulation and in networks providing BGP/MPLS VPN service, our solution also avoids packet losses during the BGP convergence that follows the deactivation of the protection tunnel, see chapter 8.

The *primary egress-secondary ingress* protection tunnels can be used to protect the interdomain links that attach providers to a multi-homed stub AS. We have proposed a simple extension to BGP that allows the routers of the stub AS to automatically advertise the parameters of the *pe-si* tunnel to be used to their provider.

Chapter 8

Graceful Shutdown of BGP sessions

Introduction

When a BGP peering link is shut down, packets can be lost even if the change is predictable or the peering link is protected with a Fast Reroute mechanism. This transient unreachability is caused by the mechanisms that are used in iBGP to disseminate routes inside Autonomous Systems. Indeed, routers can lack of alternate paths to some destinations, which makes them unable to reroute towards alternate BGP nexthops, and this, even if alternate paths have been learned by other border routers of the AS. This is unfortunate as customers often pay for multiple peering links with their providers, while providers are not able to provide a fast recovery when one of the peering links fails or is manually shut down. In [ND05], a service provider listed requirements for a mechanism that would make it possible to shut down a peering link without introducing transient loss of packets.

In this chapter, we discuss the mechanisms that can be used to improve the convergence of BGP. Firstly, we show how additional information can be disseminated inside the network to ensure that each router has previously received alternate routes to allow a fast convergence. Secondly, we discuss how a "make before break" convergence can be implemented in a network, to provide a mean to perform graceful shut down of peering links without increasing the memory load on the routers. This second solution is attractive as it requires the implementation of very few features, and does not increase the size of the routing information bases of the routers. Several options exist for the make-before-break solution, we will describe them and explain their respective trade-offs.

In section 8.1, we present the problem of transient unreachability in the case of a peering link shut down or the reception of a route withdraw. We explain how routes are propagated inside an iBGP topology and why this reduces the availability of alternate paths. The next sections cover the solutions to this problem. In section 8.3, we present techniques that can be used to improve the dissemination of alternate paths inside an Autonomous System. In section 8.4, we show how a make before break convergence can be perfomed today without implementing additional mechanisms. We also present mechanisms that can be used to cover the case where no alternate paths are present at the time of the shutdown operation.

8.1 The problem of transient unreachability

In this section, we explain how transient unreachability of external prefixes can occur when a route withdraw is received by a border router, or when an eBGP peering session is closed, and this even if alternate routes were available in some routers of the network.

8.1.1 Route propagation inside an iBGP topology

Egress routers of a domain propagate paths received from their external peers if they actually select the path. Indeed, routers only propagate their best path for each destination. An egress router that received multiple paths from a set of its peers will thus only propagate at most one of the paths. There is also a set of rules in the BGP Decision Process (DP) (local-pref, ASPath length, MED) that can force a router to prefer a path learned from an iBGP session over a path learned from an eBGP session. This impacts the knowledge of alternate paths in the network as only the egress router will know about those paths. In the context of Virtual Private Networks, the same problem can occur if the same route distinguisher is used for two distinct exit points towards a given site.

When an iBGP Full Mesh is used, each router of the network knows about as many paths towards a prefix as there are Egress Routers in the network that selected a path received from an eBGP peer. Indeed, when an Egress Router selects such a path, it will propagate it towards all its iBGP peers. A lack of alternate paths can still occur if all the routers of the network select the same Egress Router for a given prefix p.

When Route Reflectors are used, the lack of alternate paths can be worsened. For example, an Egress Router $R1$ that selected its own path towards a prefix p will only know about this path if the routers and Route Reflectors to which it is connected with an iBGP session have also selected the path via $R1$. This means that even if more than one path is selected in the network, introducing route reflectors can cause lack of alternate paths on some routers.

8.1.2 Transient Unreachability

The way iBGP works can thus lead to routers having one single route for a prefix p inside their Adj-Rib-In. This means that once this route is withdrawn, those routers will transiently be unable to select an appropriate BGP nexthop for a prefix, and will drop the corresponding packets until an alternate path has been found.

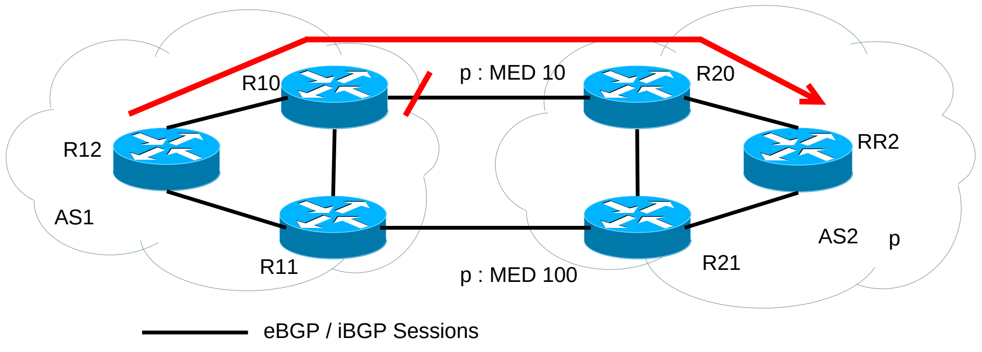


Figure 8.1: A dually connected client

To ensure the resiliency of their connectivity, neighbouring ASes are often connected with more than one link, but even in that case, some routers might lack of paths during the convergence. In Figure 8.1, we show a client AS2 that is dually connected to one provider AS1. As AS2 pays AS1 to carry packets, AS2 can use the MED attribute to force the routers of AS1 to send packets towards the prefix p via R20, which has the lowest MED[1].

Now, let us look at the propagation of BGP routes inside AS1, and let us consider that there is an iBGP full mesh in this network. $R10, R11$ and $R12$ select $R10$ as their best nexthop for prefix p. $R11$ does not advertise its own route via $R21$ because this is not its best route. Thus, we can see that $R12$ and $R10$ only know about the route via $R10$.

Let us assume that the link $R10 \leftrightarrow R20$ is manually shut down. R10 will no longer consider its route towards p and will send a withdraw towards $R12$ and $R11$. $R10$ and $R12$ will transiently consider that p is unreachable. They will recover the reachability of p only after $R11$ has removed the withdrawn route from its Adj-Rib-In, run its Decision Process, selected its route via $R21$, and finally sent it to $R12$ and $R10$. Upon the link shutdown, it is possible that a large amount of routes are in the same situation, so that the convergence process can be quite long.

Now let us consider in Figure 8.2, that the client AS2 does not use the MED attribute, and that the provider uses route reflection. $R10$ and $R11$ both select their own external path towards p, and each router sends its best path to its route reflector, $RR1$. Tie-breaking in $RR1$ will force it to select and propagate one of the two paths. Let us assume that it selects the path via $R10$. Once again, $R10$ only knows about one route for prefix p, so that a transient unreachability can occur when the link is shut down.

Policies commonly used in the Internet also have an impact on the propagation

[1]Agreements on communities can be also be used between AS1 and AS2 to achieve the same goal. By tagging a route with a community defined in the agreement, AS2 can have an impact on the local pref that will be set by AS1 to the route.

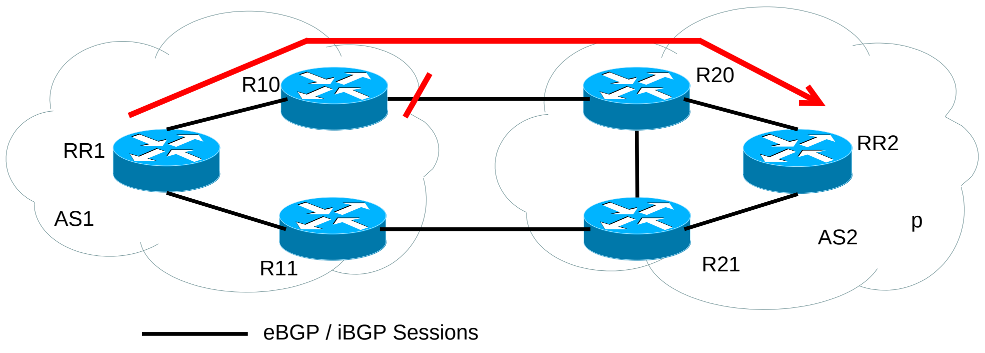


Figure 8.2: A dually connected client, without MED

of routes inside each Autonomous System. When a route is received from a client, it is always prefered over the routes received from providers or peers. This means that all those alternate routes are hidden and only known by the Egress Routers that receive them on their eBGP session.

To illustrate this, let us consider Figure 8.3. In this topology, AS2 is a customer of AS1 and AS4, which are customers of AS3. $R12$ hides the route for prefix p via $AS3$, as it prefers the client route via $R11$. If the link $R11 \leftrightarrow R21$ is shut down, $R10, R11$ and $R13$ will have no alternate path and will only recover the reachability of the prefix p after $R12$ parsed the iBGP withdraw sent by $R11$, performed a DP update, and propagated the route via $AS3$.

Note that if $R30$ had prefered the path via $AS1$, a route withdraw should have been sent towards $R30$ to let $R30$ change its decision and propagate the alternate path to $R12$. In this case, not only the iBGP topology has an impact on the convergence, as a DP update in an external router is needed. In such a scenario, alternate paths exist, but their unavailibility is not caused by iBGP route propagation inside the network. We will show in section 8.4 how to provide a graceful shutdown mechanism in such a context.

We will firstly discuss in section 8.2 the forwarding schemes that can be used in an Autonomous System, as it can have an impact on the solutions that we propose.

The firstly proposed solution is to guarantee that at least two distinct paths for each prefix are present in all the Adj-Rib-In of the routers of the AS. To do this, we can add iBGP sessions inside the topology. We will show that this does not always suffice. Sometimes, potential alternate nexthops hide their path because there is a better path in the network. That is, they do not provide an alternate path for this better one. So, we explain how propagating non best paths helps in providing alternate paths in the network. There are several means to provide this functionnality. We will review them in section 8.3.

Sometimes, an alternate path is not available at all for a set of destinations, even if a BGP convergence without loss of packets is possible. In that case, a

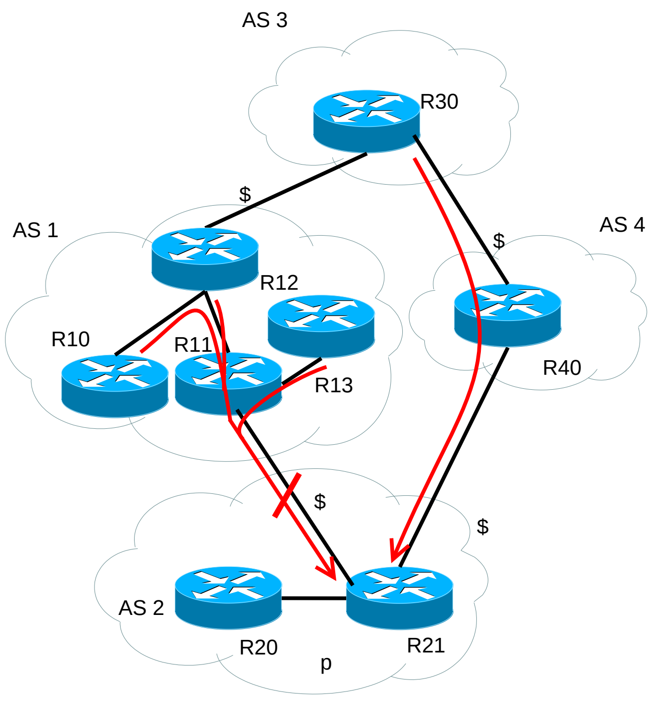


Figure 8.3: Impact of policies on route propagation

"make before break" convergence is required. Note that this kind of solution is also applicable when alternate paths are available in the network. According to this mechanism, routers first disseminate information through iBGP sessions to let routers avoid the paths being withdrawn, while allowing routers lacking of alternate paths to continue to use the initial path for a while. By doing this, routers that have alternate paths at their disposal will select and propagate them, so that all the routers of the network will finally be able to reroute, without ever being forced to drop packets. When no such paths are available inside the iBGP topology, routers will have to initiate a convergence without packet loss that involves their neighbouring peers. This will trigger advertisement of alternate paths towards the local AS. We will discuss the implementation of these make-before-break features in section 8.4.

8.2 Forwarding schemes

There are several solutions to forward a packet from an Ingress Router towards its exit peering link.

The first solution, called Pervasive BGP, lets all the routers on the path from the Ingress Router towards the exit peering link perform a lookup in its BGP table to get the BGP nexthop, and then perform a lookup in the IGP table to find the outgoing interface on which the packet must be forwarded. The first problem with such a solution is that transient (or even persistent) inconsistencies between the BGP tables of the routers on the path of the packet will let the packet finally exit the network in a different location than the one selected by the Ingress router. Those inconsistencies, called referred to as “routing deflections” [GW02b] can even lead to forwarding loops. The second problem is that all the routers on the path towards the Egress router must be BGP speakers, even if those routers are only transit routers.

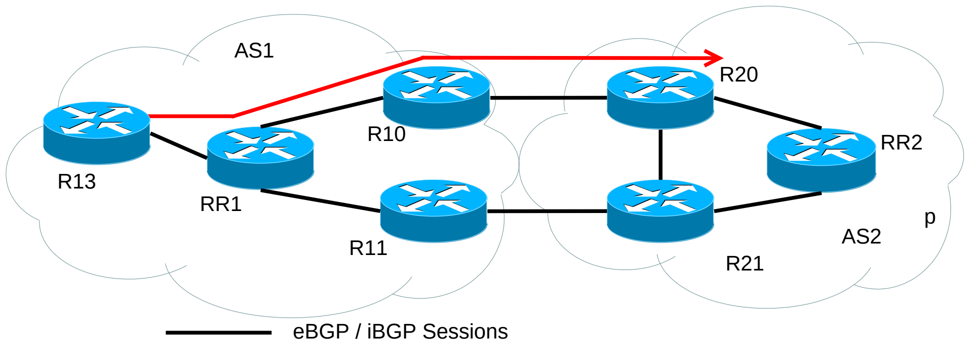


Figure 8.4: A simple topology

The second solution is to let the Ingress Router perform a BGP lookup to get the BGP nexthop for the packet, and then encapsulate the packet by using a label that will let the packet be forwarded towards the BGP nexthop [RVC01]. Routers on the path towards the Egress router do not perform lookups in their BGP table. When the Egress router receives the packet, it decapsulates it, performs a lookup in its BGP table, and forwards the packet over the corresponding peering link. When only best paths are propagated in the network, the BGP nexthop selection performed by the Ingress Router always correspond to the last external best path that the BGP nexthop selected and propagated in the network. For example, in Figure 8.4, when $R13$ receives a packet towards p, it will encapsulate the packet with a label identifying $R10$. The packet will be forwarded towards $R10$, and $R10$ will perform a BGP lookup to forward the packet towards $R20$.

The third solution is to let an Egress Router propagate each BGP path in the network with a label identifying the corresponding peering link on which the packet will be forwarded [RR99]. When an Ingress Router forwards a packet, it performs a BGP lookup to get the BGP nexthop and the label associated with the peering link of the nexthop. The Ingress then encapsulates the packet with the peering link label, and encapsulates the obtained packet with the label that it uses to reach the BGP nexthop. The routers on the path between the Ingress and the Egress will forward the packet based on the label used to reach the BGP nexthop. When the BGP nexthop receives the packet, it decapsulates it and finds the label corresponding to the peering link over which it must forward the packet. It decapsulates the packet and forwards it along this peering link. For example, in Figure 8.4, when $R13$ receives a packet towards prefix p, it will encapsulate the packet with a label identifying $R10 \rightarrow R20$, then it will encapsulate the packets with a label identifying $R10$, and forward it. When the packet arrives in $R10$, $R10$ decapsulates it, finds thanks to the second label that it must forward it along $R10 \rightarrow R20$, so that it can decapsulate the packet and forward it towards $R20$. Note that "Penultimate Hop Popping" (PHP) can be used to avoid this double decapsulation. When using PHP, the outermost label is popped by the penultimate hop, so that the BGP nexthop only has to pop the innermost label of the packet.

The main difference between the second and third solution is that the BGP nexthop does not perform a lookup in its BGP table to forward the packet. This will have an impact during the convergence, as we will see in the remainder of this chapter. In section 8.5, we show how complex it is to avoid forwarding loops during a BGP convergence when Pervasive BGP is used.

8.3 Increasing the availability of alternate paths

Having an alternate path in its Adj-Rib-In is a good way to ease fast convergence when the primary path is withdrawn. In fact, ensuring the availability of alternate paths will permit to avoid the cases where a propagation of paths is necessary to perform the recovery. As we will see in this section, having alternate paths also allows a router to adapt to a non-urgent removal of paths implied by a maintenance operation on a peering link without losing packets. When a sudden failure of a peering link occurs, and the link is protected with a the Fast Reroute technique proposed in chapter 7, the following convergence process is also non urgent and should be performed without causing packet loss.

8.3.1 Adding iBGP sessions

One possible mean to increase the availability of alternate paths is to modify the iBGP topology to guarantee that for each prefix p, each router R of the network has a session with at least one Route Reflector that performs a different route selection for p or with one Router that selects an eBGP route which is different from the route selected by R.

For example, by looking in Figure 8.2, we can see that if we add a session between $R10$ and $R11$, all the routers of AS1 will have two routes towards p. The main advantage of this solution is that it does not require any modification to the BGP protocol. What is required is a tool that will tell how to tune the iBGP topology to provide such guarantees.

Note that in this context, Egress Routers always still propagate their best paths only, so that their BGP forwarding table will be consistent with all the routes that they originated and propagated. This means that when a route is withdrawn, and an Ingress Router switches to an alternate path, the presence of the alternate path in its Adj-Rib-In implies that the BGP nexthop associated with that path has already selected the path as its best path.

Adding iBGP sessions in the network will not always suffice. For example in Figure 8.1, we have an iBGP Full-Mesh, and we still have routers with only one route for prefix p. More generally, we can say that to prepare the routers to a withdraw by ensuring that at least two routes are present, we must break the rule that routers and route reflectors can only propagate their best paths. Moreover, the tuning of the iBGP sessions will have to change in time, as route advertisements received from external peers will change. Doing this might be hard in practice because maintaining one valid and robust iBGP design is already a hard task without considering those additional constraints [GW02b].

8.3.2 Propagation of non-best paths

In this section, we explain how routers can propagate paths that are not selected as their best paths to improve the availability of alternate paths in the network.

Second best path propagation

This technique lets the routers of an AS pre-converge by considering the withdrawal of their primary path towards a given destination. This can be done by using [WRC05] or [RR01]. These protocol extensions allow BGP speakers to propagate more than one path towards a given destination. We propose to use such an extension to force routers and route reflectors to advertise their best path as well as their second best path.

The second best path of a router R for a destination d is the path that R selects, by removing the first best path from its Adj-Rib-Ins, and by adding in its Adj-Rib-Ins the second best path of its iBGP peers for which the first best path is the same as the primary path of R. Performing the second best path computation like this implies that, if the best path of a router R for a destination d is withdrawn, then the second best path that R has for d is the post-convergence primary path of the router, so that only the route withdraw is necessary to let R find its alternate path.

If a router R does not find a second best path for a destination d, by applying the mechanism described above, this means that all the iBGP peers of R have the same first best path as R and did not find a second best path for the destination. This implies that no alternate path would be found for destination d if the best path selected by R was withdrawn. In the best case, an alternate path will be found but it is not currently known inside the AS, so that a complementary solution spanning over multiple domains would be required.

Redundant paths propagation

Executing the second best path computation may require too much CPU in the routers of the network, as it forces the routers to actually perform an iBGP convergence twice for each prefix. Another solution could be to let routers propagate as alternate paths for a prefix p, the paths for the same prefix that were received from the same neighbouring AS. To do this, an Egress router can take advantage of the solution proposed in chapter 7 aimed at discovering the set S of multiple eBGP sessions with the same AS and the same policies. We will say that those sessions are redundant. When an Egress router advertises a path for a prefix p that is received from an eBGP peer, it can also advertise as alternate paths for this prefix p all the paths for p coming from sessions in S. We will call those paths redundant paths.

Compared to the previous solution, routers do not have to execute their Decision Process twice for the same destination, as alternate routes can be considered as "attributes" of the best routes. One issue is to ensure that an Egress Router has redundant paths in its Adj-Rib-In. One design guideline aimed at favouring this is to let two Egress Routers that maintain redundant eBGP sessions establish a direct iBGP session between each other, as suggested in [dSB07].

Best external route propagation

Another technique is to allow each router to propagate, inside the iBGP topology, its best external path towards each prefix. The best external path selection should respect the following rules.

1. Departing from the set of external path,
2. Remove current Best Path if external,
3. Prefer Highest Local Pref,
4. Prefer the path with Shortest AS Path attribute,
5. Prefer lowest MED,
6. Prefer lowest router address,

In the case of an iBGP full mesh, applying this scheme ensures that all the routers have an alternate path for each destination, if at least one was received at a border router of the network. However, this comes with a potentially high memory load on the routers.

When route reflectors are used, the availability of alternate paths is ensured if each route reflector propagates, towards each of its client, its best path and its second best path. Additional routes can be carried in BGP update messages using [WRC05].

Compared to the second best path propagation, we can say that sending best external paths is easier than computing the second best paths, as the best external path is selected locally, i.e. routers do not converge to find alternate paths. However, we can also say that more useless routes might be propagated in the network. For example, let us assume that in an Autonomous System, a few paths towards a given destination are available on a few client peering links. The availability of alternate paths can be ensured thanks to those client peerings. Thus, all the routers that peer with the providers of the AS, and that do not peer with a client providing one of those "client alternate paths", will propagate alternate paths in the network that are of no interest, because they have a lower local pref value and will only be selected if all the client paths become unavailable.

A route server dedicated to alternate path propagation

To improve the availability of alternate paths in the network, we can also use one (or more) dedicated route server that would be in charge of advertising alternate paths inside the network. The interest of such a solution is that it lets routers of the network do a minimal work to distribute alternate paths. The only thing they have to do is sending their best external paths to the route server, and retrieve the alternate path that is propagated by the route server. Compared to the best external route propagation, we can say that useless best external paths will only be sent towards the route server, and the route server will not propagate these towards its clients. The solution thus concentrates the increase in memory load on a device that is dedicated to this task.

Even if the route server is slow, we can say that as it is only there to prepare routers to a withdraw of their primary paths, its response time is not crucial. As the goal of this route server is only to propagate backup paths, its centralized flavor does not make it a single point of failure, as the forwarding of packets accross the network is not compromized when the route server is not functionning.

A route server can perform the best alternate path selection globally, which means that it will select one set of alternate paths for a destination and advertise it to all its clients, or it can perform an alternate path selection on a per-client basis. We will begin our analysis by describing those two possible techniques. Next, we will see how the task of the route server can be split among a set of route servers to solve the memory load issue. After that, we discuss two techniques that can be used to avoid that the propagation of alternate paths have an impact on the selection of primary best routes in the network. This last point is important as we do not want to introduce an additional potential of convergence unstability, by propagating new paths through the iBGP topology.

Per-client alternate path selection

With the **per-client** selection, each router of the network should have a session with the route server, over which it sends its current best path and its best external path, by using [WRC05] or [RR01]. The behaviour of the routers concerning their other iBGP sessions does not change.

For a given destination, the alternate path that a route server would send to a router R would respect the following rules.

Departing from all the routes known by the Route Server, for a given destination,

1. Remove the best and best external paths received from R, if any

2. Prefer Highest Local Pref

3. If any, prefer paths that are already selected as best path by their originator
   This rule only makes sense if a BGP lookup is performed by the Egress Routers when they receive packets over internal links. Its goal is to avoid inconsistencies when router switch to the alternate path while its corresponding BGP nexthop does not forward packets according to this path. Not doing this favors transient forwarding loops. To distinguish, in a BGP Update message, if a path is currently the one selected as best by its originator, the bestpath bit defined in [WRC05] can be used. When a double encapsulation forwarding scheme is used, the egress ASBR does not perform a BGP lookup to select the peering link over which the packet will be forwarded. Thus, this transient control plane inconsistency has no impact on the forwarding path selected by the Ingress Router, so that this rule is not mandatory.

4. Prefer Shortest AS Path

5. Apply MED

   The MED should be applied by considering all the available paths towards the destination. "Always-compare-med" should be configured consistently with the other routers of the network.

6. IGP tie-break

   An IGP tie-break performed from the route server point of view makes no sense as it does not consider the distance between the receiver R and the nexthop of the route. If such a tie break must be performed, the route server should compute the Shortest Path Tree of R to perform an accurate selection.

7. Prefer lowest router address

Global alternate path selection

With the **global** alternate path selection, the route server will select **a single set** of alternate paths that can be used as backup paths in the network. This set of paths must contain at least two paths with distinct BGP nexthops in order to ensure that all BGP nexthops receive alternate paths from the route server. Indeed, if the route server only propagates one alternate path, the originator of that path might lack of an alternate path.

Departing from the set of routes that the router server has for a destination, the selection of alternate paths with the global mode should respect the following rules.

1. Prefer Highest Local Pref,
2. If any, prefer paths that are already selected as best path by their originator, This rule only makes sense if a BGP lookup is performed by the Egress Routers in the network when it receives packets on internal links.
3. Prefer Shortest AS Path,
4. Apply MED,

   The MED should be applied by considering all the available paths towards the destination. "Always-compare-med" should be configured consistently with the other routers of the network.

   IGP tie-break does not make sense as the alternate path selection is performed globally.
5. Select the $< x >$ oldest routes among the remaining routes.

The value of $< x >$ should be tuned according to the number of alternate paths that is desired in the network. A route server can stop its decision process at a given step and send all the remaining routes to its peers. The step where the selection must be stopped is the step that would let the route server propagate less than $< x >$ alternate paths for the concerned destination.

Policy-driven alternate path selection

The selection of alternate paths by the route server can also be performed by focusing on the type of the primary path.

For example, if the best path for a given destination is a path received from a client, the route server can select all the client paths that it received as alternate paths. More generally, a route server could select as alternate paths for a destination, all the paths that have the same local-pref as the current best paths. If there is no path with the same local-pref, the route server could select the paths with the highest value of local-pref that can be found for the destination, or all the paths whose local pref fall into local pref range associated with all the paths from a given peering type.

Impact of alternate path propagation on primary path selection

A potential issue when alternate paths are propagated as regular paths is that, in some cases, the propagation of these could have an impact on the selection of the primary best path of a BGP router.

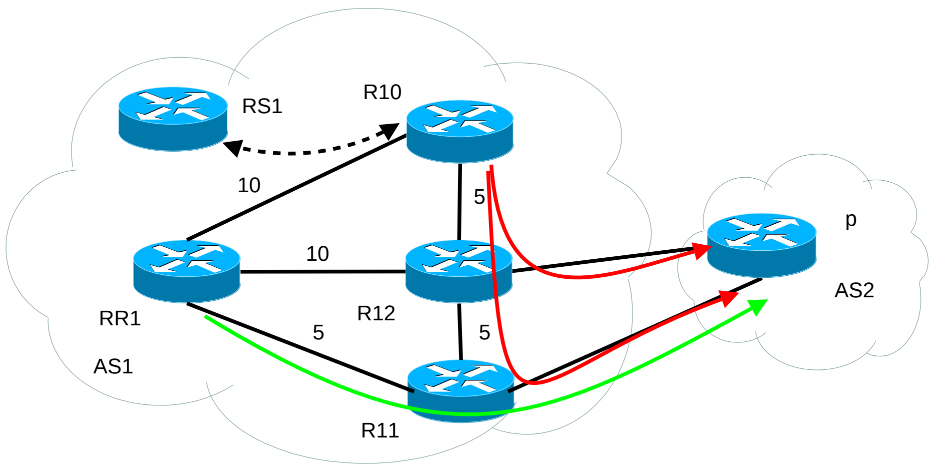


Figure 8.5: A route server

We think that this should be avoided as in some cases, the introduction of the route server itself could put the network in a routing oscillation state. To illustrate this issue, let us consider the Figure 8.5. Let us assume that the Route Server is configured to advertise only one alternate path to its clients. We can see that two paths for prefix p are available at the borders of $AS1$. One path is via $R12$ and the other via $R11$. Let us assume that $R10$ receives one path towards p via its Route Reflector $RR1$. From the perspective of $RR1$, the path via $R11$ is better when the IGP tie-break is applied. $R10$ thus only knows the path via $R11$ and selects it. The best path of $R10$ is sent towards its route server, $RS1$. $RS1$ knows about two paths, the one via $R12$ and the one via $R11$. Thus, the alternate path that $RS1$ sends towards $R10$ is the path via $R12$. But, $R10$ actually prefers this path over the path received from its RR, so that the path via $R12$ is now its best path, and $R10$ sends it to $RS1$. At that moment, $RS1$ must adapt and send another alternate path towards $R10$, which will be the path via $R11$. $R10$ now only knows about the path via $R11$, that it receives from its RR and from its RS. It will send its best route towards the route server, and one cycle of the oscillation has been completed.

The first solution to this problem is to let the route server send alternate paths with a local-pref value of 1. However, this has a negative impact as it could force one router receiving a route withdraw for its primary path to select one of its external paths via a provider, although the alternate path that was sent by its route server is a valid path via a client. We suggest to solve this problem by taking advantage of the gaps that ISPs usually leave between the different values of Local-Pref assigned to their routes. Setting the local-pref of routes such that the difference between the local-pref of two routes is 0 or at least 10, will help in degrading the alternate paths sufficiently to prevent them from impacting the selection of primary path, while degrading them not too much to favor an as direct as possible transition to the best alternate paths upon convergence. By doing this, the route server can propagate its alternate path by setting its local pref value to the local pref value set by the its orginator, minus 5. That is, alternate paths will never be preferred over the best paths that are learned on normal iBGP sessions, while maintaining the respect of the policies when an alternate path has to be selected.

With the local-pref shift proposed above, $RS1$ would have reduced the local pref of the path via $R11$, so that $R10$ would not have changed its best path upon reception of the path via $R11$ from its Route Server.

Choosing one solution

We proposed several solutions relying on the propagation of non best paths. Selecting one best solution is not an easy task as they will depend on multiple factors such as the memory load overhead that the BGP routers of the AS can afford, as well as the cpu load overhead. Also, the number of alternate paths available at the borders of the AS should play a role in this decision.

However, solutions relying on route servers precisely offer ways to deal with these constraints in a flexible fashion. Indeed, the number of alternate paths propagated by the route server can be tuned according the available memory in their clients. The cpu load is concentrated on the route server, and could be provisionned accordingly. Also a peak in the cpu load of the route server would not harm the availability of primary paths among the other BGP routers of the AS.

8.4 Make before break iBGP convergence

In the previous sections, we discussed ways to let routers advertise more paths over iBGP sessions, in order to increase the availability of alternate paths upon a shutdown. All those solutions unfortunately increase the RIB memory consumptions of the routers. Also, these solutions are not available yet.

In this section, we present another mechanism to perform a convergence that anticipates the maintenance of a peering link without loosing packets. This solution does not suffer from these issues.

We firstly present the solution when a shutdown of a Customer-Provider link is performed at the provider side. We tackle the problem of avoiding packet loss for the outgoing traffic (from the provider to the customer) and for the incoming traffic. Then, we discuss the variant when the shutdown is performed at the customer side. It is to be noticed that in all these cases, we assumed that there is at least one peering link to backup the peering link being shut down. In the last part of this section, the case where this property is not verified is examined and a solution to them is proposed.

As a bonus, we will see that the proposed solutions help in concealing the convergence, and thus the BGP path exploration as locally as possible, so that the

BGP churn is reduced.

The idea underlying the scheme is the following. Firstly, a link remains up while routers adapt to its scheduled removal. As we will see, this is not sufficient to avoid all packet losses, because some BGP routers can lack of alternate paths. Thus routers are also allowed to keep using the paths via the link being shut down until they find alternate ones. These compromised paths will be made less preferable, so that the convergence process will take place and alternate paths will be spread across the network.

8.4.1 Shutting down a $Provider \rightarrow Customer$ link

Let us assume that a BGP peering link is shut down. This link goes between an internal router PE and a client router CE of a client AS. To avoid packet loss, two problems have to be solved. The first one is to ensure that the routers inside the local AS stop using the paths towards the destinations that were reached via the $PE \rightarrow CE$ link without loosing packets. The second one is to ensure that the routers behind CE stop using the paths for the destinations that they used to reach via $CE \rightarrow PE$ without loosing packets. As a consequence the packet flow must be uninterrupted in both directions thus preserving the reachability during the convergence.

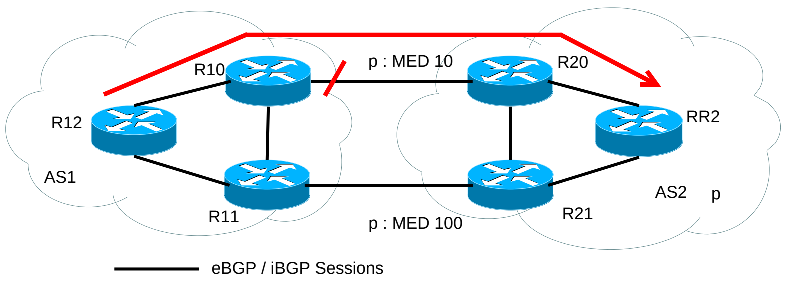


Figure 8.6: A dually connected client

Outgoing problem

The simple topology shown in Figure 8.6 will be used to understand the problem. In this topology, there is a full mesh of iBGP sessions among all the routers. We assume that the link between $R10$ and $R20$ will be shut down by an operator of AS1, a provider of AS2. Let us assume that $R10$ keeps the link up for a while, and behaves as if the link was down. Consequently, $R10$ sends withdraw messages to its iBGP peers for the paths received from $R20$ that it selected as its best paths. The recovery will be performed by using the paths along $R11 \rightarrow R21$.

A transient unreachability for a prefix p may still occur because routers may have only known the route for p via $R10 \rightarrow R20$ in their Adj-Rib-In. Thus they will drop packets destined to p until they receive the alternate path. For example, if there is an agreement between AS1 and AS2 to allow AS2 to perform incoming traffic engineering by using communities or MED, $R11$ may prefer the path via $R10 \rightarrow R20$ over the path received on its own eBGP peering link with $R21$, so that the secondary path will not be propagated towards $R12$ and $R10$. When $R12$ receives the withdraw from $R10$ for the path towards p, it starts dropping packets for this prefix. When $R11$ receives a withdraw for a prefix p from $R10$, it selects its external route for p and propagate it on its iBGP sessions. The recovery will only be accomplished when $R12$ receives the update from $R11$.

This example illustrates that ensuring the forwarding along the link being shut down is not sufficient to provide a packet loss-free convergence. In addition, routers should avoid the paths that will become invalid, as soon as possible, while allowing those routers to still use these paths if they do not have alternate paths. To do that without modifying BGP, we must use a two-step approach. First, we must render the affected paths less preferable than any other available path. Thus, the attributes of those paths must be modified to impact their quality at the very first step of the BGP decision process, and let routers select an available secondary path. Second, the link must be actually shutdown and the obsolete paths must be withdrawn.

To perform the first step, we could set the local-pref attribute of those paths to 0, and let the router performing the shutdown advertise updates for those paths. In the example above, $R10$ should do this. After a while, the other routers will have switched to the alternate paths. At this time, $R10$ is allowed to withdraw the old paths. This operation will have no impact on the forwarding since those paths are no longer used to forward packets.

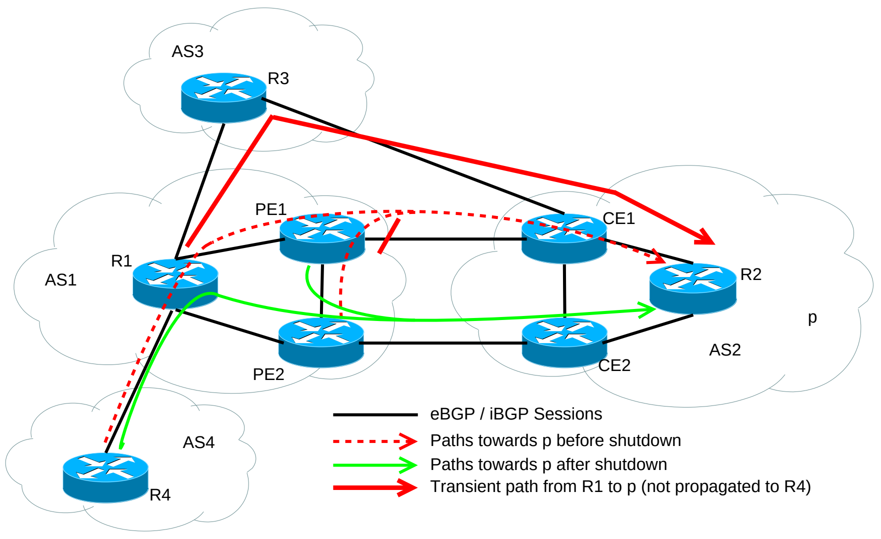

AS3
R3
PE1
CE1
R1
AS1
R2
p
AS2
PE2
CE2
AS4
R4
eBGP / iBGP Sessions
Paths towards p before shutdown
Paths towards p after shutdown
Transient path from R1 to p (not propagated to R4)

Figure 8.7: A dually connected client

This solution is adequate in the context of VPN, where the possible alternate paths always come from the same neighbouring ASs. It also works in the case where the client is a single-homed AS that is dually connected to its provider.

However, the solution is not safe enough in the case of regular internet traffic, where some alternate paths for client paths could be advertised by providers. The topology depicted in Figure 8.7 will be used to illustrate this problem. In this case, AS3 is a provider of AS1, and AS4 is a shared-cost peer of AS1.

During the first step, $PE1$ will send an update with a local-pref attribute of 0 towards $R1$ for prefix p. For the same reason as described in the previous example, $R1$ may not know about the path via $PE2 \rightarrow CE2$. In this case, $R1$ will select an alternate path via $R1 \rightarrow R3$, which would be the only one with an higher local pref value. But according to the usual peering relationships among neighbouring ASs, this provider route cannot be advertised to the providers or shared cost peers of AS1. Consequently, $R1$ will send a withdraw to $R4$ for prefix p and the recovery will only be performed once R receives the alternate client path, selects it as its best route, and sends an update to $R4$. Meanwhile $R4$ will transiently drop the packets destined to p, and this unreachability could be propagated through the Internet.

A local-pref of 0 cannot always be used in the first step. However, a local pref that is lower than any local pref assigned to client routes within the ISP and higher than any local-pref assigned to the routes receives from providers and shared cost peers can be used. As a result routers will keep using the obsolete path until new client paths are advertised. Thus only update messages will be advertised to providers and shared cost peers, instead of abrupt withdraws.

Incoming problem

The routers on the neighbouring AS must stop using the link being shutdown. In the example topology of Figure 8.6, the routers in AS2 should stop using routes passing through $R20 \rightarrow R10$.

The first solution is basically to contact the operators of $AS2$ and let them use the same technique as described in the preceding section. Although it works, it is not very convenient because it requires synchronizing operating teams. Furthermore, maintenance is generally performed during the least disturbing time periods for the client (during the night for example). The maintenance operation may require the client to assign dedicated human resources for this task, which is unfortunate. It may also induce additional financial cost for the provider and the client. Moreover this task becomes a real scheduling nightmare when the maintenance affects multiple clients at a time, e.g. in the case of a linecard removal or a complete shutdown for a typical provider edge router.

A simpler solution is to have the provider agree with each client on a BGP community that would be dedicated to routes that have to be avoided by the client. When the provider performs the shutdown, it will re-advertise its paths by tagging them with this community. On the client side, the routers will have been pre-configured to set a local-pref value of 0[2] to all the routes tagged with this community. After a while, the router on the provider side will send a Cease Notification to actually withdraw the routes and shut down the session.

This solution is applicable without modifying BGP. However, is not very fast as it requires $R20$ to re-advertise all its paths towards $R10$.

Another solution is to implement a new BGP message which would simply mean that the session will be shut down within a given amount of time, and that the CE should adapt to it by using the techniques described above. This message can be for instance an eBGP Cease Notification message with a new sub code or a new dynamic capability.

8.4.2 Shutting down a $Customer \rightarrow Provider$ link

When a $Customer \rightarrow Provider$ peering link is shut down at the customer side, the proposed behaviour of the routers is similar to the one proposed when the provider performs the shutdown. We will thus briefly summarize the behaviour that should be applied.

Outgoing problem

When a peering link between a customer and its provider is shut down at the customer side, the router where the shutdown command is issued must set a local pref of 0 to the routes that it received over the impacted link, in order to reroute the traffic that was going from the Customer towards the Provider. This will force routers on the customer side to select paths received over other peering links with providers.

Incoming problem

When the graceful shutdown is performed by using an agreement between the customer and the provider, the local-pref value that has to be set by the provider must be lower than any local-pref value assigned to client routes inside the ISP, and higher than any local-pref value assigned to routes received from providers and shared cost peers. This will force the routers on the provider side to select paths via other peering links with clients. After a while, the local-pref value will be set to 0 to let routers of the provider select alternate paths via other peering links (shared-cost or provider peering links), for the prefixes for which no alternate paths via customers could be found.

8.4.3 Shutting down a Shared-Cost peering Link

The simpler solution in the case of a Shared Cost peering link shutdown is to also set a local-pref value of 0 on the routes received over this link to solve the **outgoing**

[2]According to the usual peering relationships, using a local pref value of 0 will not trigger the sending of path withdraw messages

problem, and to also use an agreement on a dedicated community to solve the **incoming problem**.

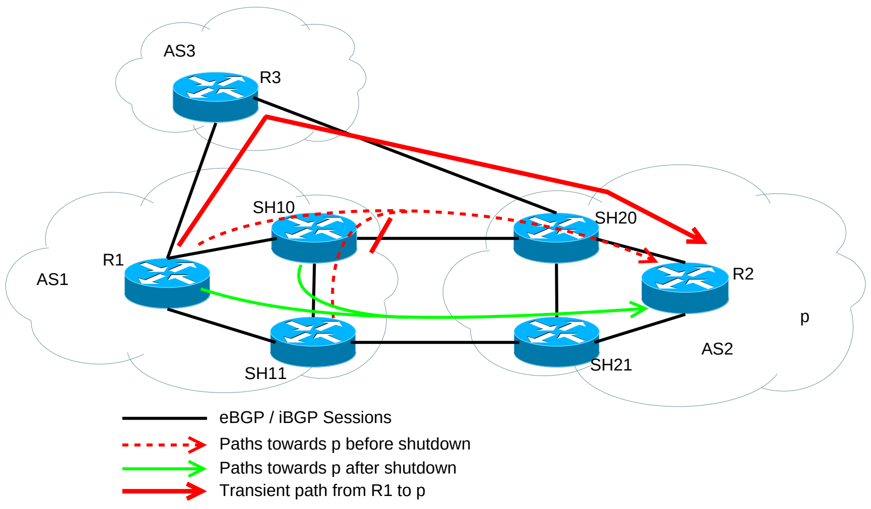

AS3
R3
SH10
SH20
R1
AS1
R2
p
AS2
SH11
SH21
eBGP / iBGP Sessions
Paths towards p before shutdown
Paths towards p after shutdown
Transient path from R1 to p

Figure 8.8: Dually connected Shared-Cost peers

However, this could cause a transient utilization of provider links to reroute the traffic even if, after the convergence of the network, alternate paths through other Shared-Cost links will be used. For example, in Figure 8.8, let us assume that AS1 and AS2 have a Shared Cost peering relationship. Once again, it is possible that $SH11$ prefers a path via $SH10 \rightarrow SH20$ to reach p, so that it does not advertise its own alternate path to AS1. When the link between $SH10$ and $SH20$ is shut down on $SH10$, $SH10$ would send local pref updates to 0 for its path towards p. $R1$ would then select the only alternate path that it knows at that time, which is a provider path via AS3. Finally, when $SH11$ selects and advertise its own path via $SH11 \rightarrow SH21$, $R1$ will prefer this path via this Shared Cost peering link and reroute again.

This scenario does not lead to packet losses as for the Provider-Customer link shutdown case. Indeed, according to the usual peering relationship model, routes received over shared cost links or links with providers are only advertised to clients. Thus, a router switching from one kind of route to the other will not send BGP withdraw messages to its peers. However, the operators of AS1 might not want to transiently use alternate paths via providers if paths via shared cost peerings are available. The reassignment of the local-pref to the routes becoming invalid should then be done with a value that is lower than all the ones assigned to Shared-Cost routes, but higher than the local-pref values assigned to provider routes. As a second step, a local-pref value of 0 should be re-assigned to those routes to face the case where no alternate path can be found for some prefixes over other Shared-Cost peering links.

8.4.4 The single link shutdown case

When there is only one peering link between two ASs, it is not ensured that the reachability can be recovered locally to the two ASs, upon a shutdown. This means that, even if there will not be unreachable destinations after the convergence, there are some cases where no alternate path could be directly available at the borders of the local AS at the moment of the shutdown. Even though the clients that care about the convergence time in their network tend to connect with multiple links to their providers, there are some cases where the redundancy could be obtained from different providers. For example, we could easily imagine the dual-homed stub scenario depicted in Figure 8.9.

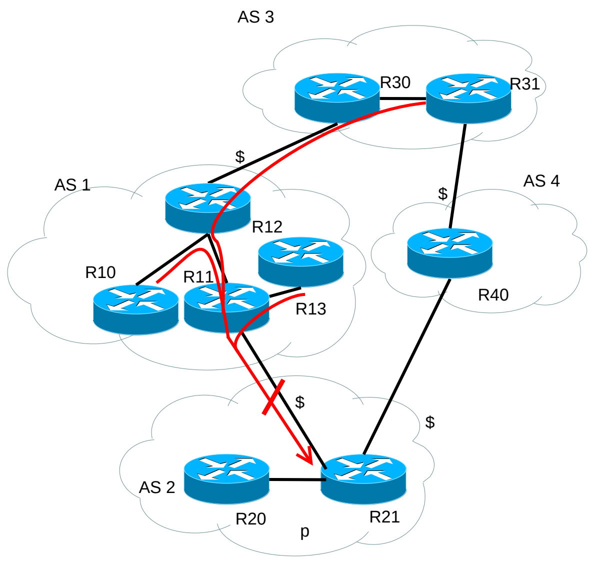


Figure 8.9: A dual-homed Stub

In that scenario, $AS2$ buys Internet connectivity from two providers, $AS4$ and $AS1$, and is connected with a single link to each provider. Now let us assume that the link between $R11$ and $R21$ undergoes a maintenance operation. Before the shutdown, $R30$ in $AS3$ reaches prefix p advertised by $AS2$ via $AS1$, and we can easily imagine that $R31$ does the same, for traffic engineering reasons. If a local-pref value of 0 is assigned inside $AS1$ on the routes learned over its peering link with $AS2$, this will not affect the selection of the best path for p because this is the only known path in $AS1$.

To recover the reachability of p in $AS1$, $R30$ in $AS3$ must be notifed with a withdraw from $R12$ that its path to p via $AS1$ is not valid anymore. The withdraw would be propagated to $R31$ which would then select its alternate path via $AS4$, and then propagate it towards $R30$. Finally $R30$ would advertise this path to $AS1$. If the convergence is done in the regular way, packets to p will be dropped by $R30$

between the reception of the withdraw from $R12$ and the reception of the path update from $R31$.

If $AS1$ and $AS3$ have an agreement on some dedicated communities to perform graceful shutdown of their peering links, $R12$ could take advantage of it. $R12$ could tag the paths for which it cannot find an alternate with this community, and propagate an update to $R30$. This will notify $R30$ that those paths will become invalid within some time.

If there is no agreement between $AS1$ and $AS3$, then the normal convergence process will take place when the BGP withdraw messages will be received by $R12$. These withdraw messages will be propagated upon the actual shutdown of the peering link. That means that even though the initiation of the recovery using withdraw messages will be delayed, the LoC will not last longer than if nothing had been tried to preserve the reachability of the affected prefixes during the process.

If such a convergence process without packet loss is required to spread through multiple ASes like in this scenario, we would recommend to implement the solution using transitive BGP path communities, as a transitive eBGP cease notification message is meaningless. Indeed, even though a cease notification message would be meaningful at the location of the shutdown, e.g., on $R21 \leftrightarrow R11$, remote ASBRs must only propagate information about the obsolescence of the best paths via this particular peering link, and not the other paths, so that in this case the information must be transmitted on a per prefix basis.

There is a tradeoff between the convergence concealment brought by the solution, and the opportunity to span the convergence without packet loss across multiple ASes. If an AS $AS1$ transits the “obsolete’ communities beyond its borders, then a shutdown initiated by one of its neighboring AS, $AS2$ might let $AS1$ propagate updates although the convergence could have been concealed between $AS1$ and $AS2$.

A way to solve this problem is to use **cascades of standardized communities** and define relations between them. We could for example standardize 5 values of communities that would be used to tag obsolete paths. The values, say $Obsolete\_Path\_Community_i$ (OPC_i), with $0 \leq i < 5$. The transitivity of such community would be defined as follows : OPC_0 is not transitive. A path which is tagged with a community OPC_j, must be tagged with the community OPC_{j-1} when advertised over an eBGP session, and the community OPC_j must be removed from this path.

With this technique, the operator performing the shutdown could shut its link down by attempting to conceal the convergence at most, using OPC_0. If traffic continues to flow along the peering link undergoing the maintenance after sometime, this means that the neighboring AS does not find an alternate path, and hence should start exploring beyond its borders.

Such cascades can also be used to progressively control the type of peering link over which alternate paths can be selected.

8.4.5 iBGP Aggregate Graceful Withdraws

When a peering link is shutdown, all the routes that were received on this peering link will have to be removed from the Adj-Rib-Ins of all the routers of the AS. In order to perform a faster convergence, a new kind of BGP message can be used to specify that all the routes received from this peering link must be removed.

Aggregate Withdraws were defined in [RPAW05] to be able to withdraw a set of routes sharing a property. Typically, the property is the peering link being shut down. To detect that a route was received from a given peering link, communities can be used [RPAW05]. If next-hop-self is not used in the network, the next-hop can be used as it identifies the peering link by itself. The solution can be used to gracefully remove sets of VPN routes if different route distinguishers are used for all the routes.

In the normal BGP case or in the context of VPN without distinct route distinguishers, the issue of transient unreachability must be solved, so that we have to perform an "Aggregate Set-Local-Pref 0" or an "Aggregate Withdraw" by using the $TIME\_TO\_WITHDRAW$ feature. This must be done to allow routers that only have the route being withdrawn for a given destination to receive alternate routes before updating their FIB, so that the destination will not be transiently considered as unreachable.

When a router receives an aggregate graceful withdraw specifying a given property, it must set the local-pref of all the routes verifying this property to 0, and propagate the aggregate graceful withdraw. This will force routers and route reflectors to select and propagate alternate routes. After a while, an aggregate graceful withdraw can be propagated, using the same property, in order to remove the routes (whose local-pref was updated to 0) from the Adj-Rib-Ins of the routers that still have it.

We can also specify a $TIME\_TO\_WITHDRAW$ in the aggregate graceful withdraw in order to force the receiving router to reflect, in its FIB, a potential lack of BGP route in its RIB only after the specified time. In this case, it is not necessary to flood an additional aggregate withdraw message. The value of $TIME\_TO\_WITHDRAW$ must be defined according to the number of routes that are concerned by the withdraw.

8.5 ASes using Pervasive BGP

In autonomous systems using pervasive BGP, the solution described above can lead to forwarding loops. The main problem in such networks is that each iBGP message that causes a change in the FIB of one router may cause a transient forwarding loop. Such forwarding loops have been detected in large ISP networks [HMMD02].

To illustrate the problem, let us consider again the topology shown in figure 7.10. If `AS1` is using pervasive BGP and we modify the primary egress router to send an iBGP update with the `local-pref` attribute set to 0 then destination

1st BGP message	2nd BGP message	3rd BGP message	4th BGP message	Comment
$R2 : U^0_{R1}$	$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R2 : U^0_{R1}$	$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R2 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R3 : U^0_{R1}$	$R2 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R2 : U^0_{R1}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	$R2 : U^0_{R1}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U^0_{R1}$	$R2 : U_{R3}$	transient loop `R1-R2` between second and fourth message
$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	$R2 : U^0_{R1}$	transient loop `R1-R2` between second and fourth message

Table 8.1: Transient loops caused by the updates of the FIBs with pervasive BGP

D remains reachable. However, during the iBGP convergence, the ordering of the updates of the FIBs is important. In table 8.1, we summarise what happens during the eight possible orderings of the FIB updates. In this table, $Rx : U^0_{Ry}$ indicates that router Rx has updated its FIB after the arrival of the iBGP messages with `local-pref` set to 0. Out of the eight possible orderings, only three are always loop-free.

Avoiding transient loops in autonomous systems using pervasive BGP is a difficult problem.

Firstly, a loopfree pervasive BGP convergence must be performed by respecting orderings on a per prefix basis. Indeed, when the `MED` attribute is used, the BGP decision process does not result from a lexicographical ordering [GW02a]. As a consequence, a router R may need to change its BGP nexthop for a destination $p2$, as a result of the failure of $R1 - X1$, even if its current best BGP path towards $p2$ was not via this link. This will be the case if the path to $p2$ via $R1 - X1$ is the path received from ASx with the best MED value, and it is not selected by R. This happens when there is a closer BGP Nexthop for $p2$, via another neighbouring AS, let us say ASz, and the second best MED path for $p2$ via ASx is closer from R than the firstly selected route via ASz. R will then change its FIB by selecting this path as it becomes the MED best route to $p2$ from ASx.

To illustrate why this absence of a lexicographic ordering would lead to a per prefix ordering, let us consider the topology of figure 8.10, where the link $R1-X1$ fails. On this link, $AS1$ received the MED best routes from ASx for destinations $p1$ and $p2$. The second ASx MED best route for $p1$ is received via $R2-X2$, and the second ASx MED best route for $p2$ is received via $R6-X6$. $AS1$ receives a route to $p2$ on the link $R5-Z1$. $R3$ and $R4$ will select this route due to the IGP tie-break in the route selection for $p2$.

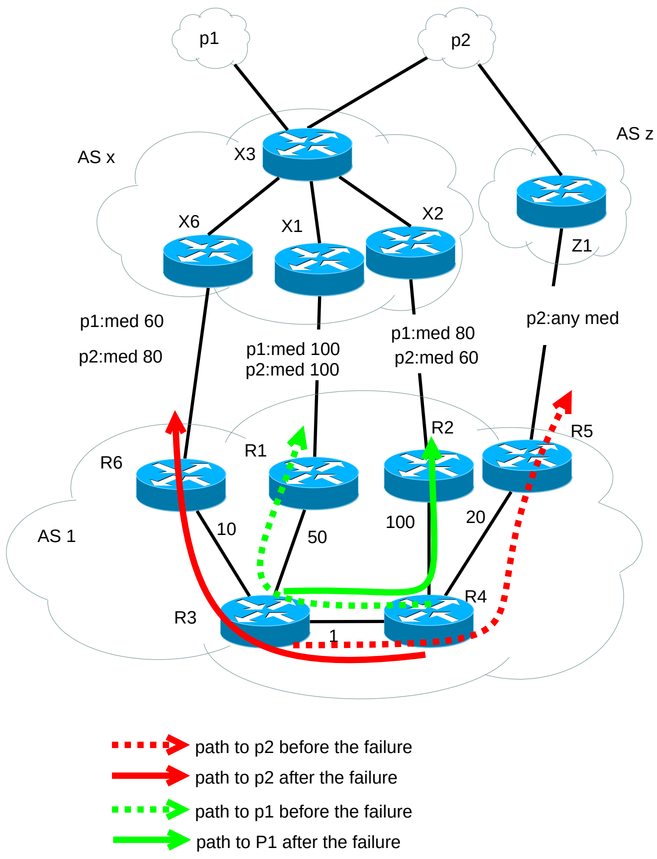

p1
p2
AS x
X3
AS z
X6
X1
X2
Z1
p1:med 60
p2:med 80
p1:med 100
p2:med 100
p1:med 80
p2:med 60
p2:any med
R6
R1
R2
R5
AS 1
10
50
100
20
R3
R4
1
path to p2 before the failure
path to p2 after the failure
path to p1 before the failure
path to P1 after the failure

Figure 8.10: $R3-R4$ conflictual update ordering with MED

When $R1-X1$ fails, $R3$ and $R4$ will update their FIB for destinations $p1$ and $p2$. Unfortunately, respecting the order that is necessary for destination $p1$, actually reached via $R1-X1$, will cause a forwarding loop for destination $p2$ on the link $R3-R4$. In fact, $R3$ is going to forward packets with destination $p1$ to $R2-X2$, via $R4$. But as $R4$ was reaching $p1$ via $R1 - X1$ and $R3$, $R4$ must update its FIB for destination $p1$ before $R3$. $R4$ is indeed a child of $R3$ in the tree composed by the paths used to reach $R1 - X1$. But if $R4$ updates its FIB before $R3$, $R4$ will firstly decide that $p2$ should now be reached via $R6 - X6$. As $R4$ is using $R3$ to reach $R6 - X6$, packets with destination $p2$ will transiently loop between $R4$ and $R3$, as $R3$ is still forwarding packets with destination $p2$ towards $R5 - Z1$, via $R4$.

BGP cannot afford such a constraint given the number of destinations that can be affected by the failure of a single link.

Secondly, ensuring routers forwarding consistency for a prefix, from an Ingress to its selected Egress point would force routers to perform their FIB updates based on the same knowledge of the alternate paths towards this prefix. iBGP dynamics are far from ensuring this property, so that complementary mechanisms, e.g., route servers, would have to be used to populate the rib-in of the routers to achieve this goal.

To conclude this discussion, we think that solving this transient inconsistency problem when routers use Pervasive BGP would require a much more complex solution than deploying an encapsulation technique, which brings other gains. So, to us, it is not worth the effort to push such solutions forward.

8.6 Conclusion

In this chapter, we tackled the problem of shutting down an eBGP peering link without loosing packets. We have analysed the aspects of iBGP that lead to transient unreachabilities in the case of the shutdown of an eBGP peering link.

Three main ways to solve this problem have been investigated.

The first one is to establish additional iBGP sessions to increase the availability of alternate paths in the routers.

The main disadvantage of such a solution is that it requires a thorough monitoring and analysis of an Autonomous System, and a responsive mechanism to adapt to changes in the set of routes received at the AS borders. Moreover, there are many cases where establishing more iBGP sessions cannot help in achieving the goal.

The second solution is to change iBGP to ensure the availability of alternate paths across the domain. This can be done by introducing features like "BGP external best" and BGP multiple paths advertisement. Currently, internet-drafts suggesting to advertise multiple paths over a BGP session do not define how to select the paths that should be advertised. They rather define the changes in the BGP update message format that should be used. We investigated different ways to select the external best path to be propagated and to select the set of alternate paths that a router would propagate to its peers. A combination of such features with route servers can be performed to increase the availability of alternate paths while concentrating the work load on dedicated platforms. Hence, routers would not risk to waste resources in selecting backup paths while an urgent convergence on primary paths is required.

This second type of solution is attractive, but increases the memory usage of BGP tables, which could be an issue in currently deployed routers.

Thirdly, we investigated ways to perform a graceful shutdown of a link by initiating a make-before-break convergence, ensuring that routers lacking of alternate paths are able to use the obsolete ones until they receive alternate paths. One implementation of this solution can be done only using router reconfiguration, so that the solution can already be applied by ISPs. We also examined ways to implement the solution in the routers themselves, and ISPs would benefit a lot from them, sparing management costs and preventing reconfiguration mistakes. Standardization could also help as it would reduce the configurations that are required to apply the solution.

We think that what has been proposed in this chapter is really interesting for an ISP as maintenance operations are common events in ISP networks. Also, the solutions described here can be applied to let BGP converge without packet losses after the application of a Fast Reroute scheme used to protect eBGP peering links.

Chapter 9

Conclusion

This thesis was aimed at improving the reactivity to topology changes of intradomain and interdomain unicast routing protocols. Its motivation is that IP routing protocols do not provide guarantees on the convergence time in case of failure. That is, the reactivity of IP routing protocols to a failure depends on characteristics of the topology such as its shape and size. The TCP/IP protocol suite has been designed with a best effort perspective, and the convergence time of routing protocols has not been a concern for years. But the emergence of applications like Voice and Video over IP, online games, and the use of IP to transmit real time and mission critical information shifted the Quality of Service requirements to more ambitious levels.

As a consequence, Internet Service Providers (ISPs) face more and more stringent Service Level Agreements (SLAs) in terms of service restoration time. From a business perspective, a stringent SLA is not a problem as customers have to pay a higher price. However, current IP routing protocols do not allow ISPs to guarantee a fast restoration of service at a low cost. Another concern for the ISPs is that maintenance operations unfortunately lead to service disruption despite their predictable nature, and can jeopardize the respect of such SLAs.

As a first approach, we evaluated in chapter 2 the limits of current link-state **intradomain** routing protocols, also called Interior Gateway Protocols (IGP). The analysis was carried out by using white-box measurements performed on Cisco 12000 routers as the input for simulations of the convergence of IS-IS in large ISP networks. From this chapter we learned that even though a fast intradomain convergence can be achieved, the number of prefixes advertised in the IGP and the way ISPs design their topologies do not easily allow a sub-50 msec convergence. We also proposed some recommendations to achieve the shortest possible convergence time with IS-IS.

At the **interdomain** level, the regular convergence process of BGP renders a sub-50 msec target completely unrealistic, even when alternate paths are available close to the failure. We detailed several reasons for such a long convergence of BGP in section 8.1.2. It actually takes seconds for BGP to recover the reachability of affected destinations, so that additional schemes are absolutely required for BGP.

Based on the fact that both intra domain and inter domain routing protocols cannot easily provide a fast recovery in case of failure, this thesis investigated ways to work around this problem.

The **first building block** of this thesis is to provide a local restoration in case of failure, triggered by a "nearly atomic" operation. That is, we had to allow routers to locally recover the forwarding of packets as soon as they detect the failure of an adjacent link or router. In other words, the recovery should not require a distributed component, like for example the dissemination of information about the failure throughout the network. Also, routers must be able to trigger this local restoration by performing an operation whose duration is fixed.

Solutions already exist at the **intradomain level**, using MPLS fast reroute tunnels established with RSVP, and their pure IP counterparts are currently under investigation by the IETF. In chapter 3 we evaluated these pure IP Fast Reroute techniques on different ISP topologies. What we learned from this study is that among the proposed solutions, the simplest, most scalable ones are not able to protect all packet flows from the failure of links in real topologies, and the importance of this issue depends on the shape of the topology graph. On the other hand, the only solution providing 100% protection coverage of links and nodes (NotVia) is a computationally expensive solution that does not scale very well. As a conclusion, we argue in favor of a combination of Loop Free Alernates (LFAs) and NotVia, to provide a lightweight protection for the links and nodes whose surroundings allow the utilization of LFAs, and enable the heavy machinery of NotVia only when LFAs cannot apply.

At the beginning of this work, no Fast Reroute solution existed to protect **interdomain** BGP peering links from failures. In chapter 7, we learned from an analysis of ISP data that such failures were as frequent as intra domain link failures, and most of them were short-lived. We proposed a fast reroute solution relying on slight modifications of BGP, which is capable of letting redundant peering links between transit ISPs used to protect each other. We also proposed a solution allowing stub ASes to quickly protect a peering link with a provider by using a peering link with another provider.

When a local restoration is performed, the flows of IP packets affected by this restoration do not follow optimal end-to-end paths. That is, these packets do not follow the paths that they would have followed if a normal convergence of the routing protocols had been initiated after the failure. We argue that when the considered failure is short-lived, this is not a problem as the optimal end-to-end paths will be used again once the failing element is brought back to service. However, when the failure lasts long, this transient mismatch should be solved and the network should be allowed to use the post-convergence optimal paths across the network.

This is where the **second building block** of this thesis finds its place. We argue that a convergence to the new optimal paths w.r.t. a failure should be performed without packet loss. Also, when the event triggering a convergence is not a failure but a topological change due to a maintenance operation, the packet forwarding service should not be disrupted.

In chapters 5 and 6, we proposed two different solutions to this problem, for link-state **intradomain** routing protocols.

The first one, “ordered FIB updates”, relies on a slight modification of IS-IS and OSPF. The solution orders the updates of the FIB of the routers by ensuring their consistency during the convergence phase. We proposed the solution, proved its correctness, and studied by simulation the time required to transit from the initial forwarding state to the optimal one without loosing packets. We conclude that even in large ISP topologies, the overhead is marginal compared to the convergence time obtained with the normal, non loop free, convergence process.

The second solution that we proposed is based on a progressive reconfiguration of the metrics of the links whose state is modified by a maintenance operation. Its strength lies in the fact that no standardization is required to apply it, as it does not involve changes in the IS-IS or OSPF protocol. In the absence of a faster, protocol built-in solution, this second technique is the only solution that a provider could use today to avoid forwarding disruptions when it reconfigures its internal topology.

In chapter 8, we proposed a simple make-before-break solution to be applied when an **interdomain** peering link is shut down by an operator. We also proposed different flavors of this solution. The first one is based on reconfiguration of BGP routers to be performed before the maintenance, in order to avoid packet losses due to a lack of alternate paths in the BGP routers. Thanks to this reconfiguration prior to the shutdown, routers will be able to keep using the obsolete paths until the reception of alternate paths from their peers. A fully automatic solution has also been proposed, and can be used to avoid packet losses during the convergence following the activation of a BGP Fast Reroute protection tunnel. This second solution requires standardization.

Perspectives

The study of the convergence time of IS-IS is already obsolete as improvements are continuously brought to both software and hardware of routers. An evaluation of routers performances regarding IS-IS and OSPF and a study of their impact on the convergence time in large networks should be continuously performed as new software and hardware are released.

Additional aspects of the IP Fast Reroute suite for the IGP should be studied. It would be interesting to analyze the impact of such techniques on the traffic, and compare them with the solutions not considered by the IETF like Multiple Routing Configurations and Failure Insensitive Routing. Also, we need to study the impact of such solutions on the provisionning applied by ISPs.

We need to study more extensively the applicability of our BGP Fast Reroute solution w.r.t. the policies that are applied on redundant peering links between neighboring ASes. We also have to investigate with ISPs if this solution is sufficient, or if more complex solutions are required.

For the loop avoidance scheme relying on metric increments, we intend to analyze the applicability of this technique as a support for loop free convergence after the sudden failure of a protected link. To do this, we have to study potential implementations of the solution in the router themselves. Also, we could study the metric sequences by only considering the destinations that are prioritized by an operator. Indeed, it is likely that long metric sequences are due to destinations that are not important for the ISP. Using data on prioritized destinations, we could probably further reduce the length of those sequences while protecting important destinations like nodes tracking prefixes related to VoIP gateways.

The graceful shutdown mechanisms for eBGP peering links would need to be experimented in real environments, and we need to interview more ISPs to decide if the gain provided by the standardized solution is worth the engineering effort.

A major topic that was not considered in this thesis is Multicast. At the time of this writing, we traced the main lines of a Fast Reroute solution for Single Source IP Multicast, as well as a graceful shutdown mechanism. As a further work, we need to publish those solutions and get feedback on them from the research community. Also, we would like to investigate potential loop storms in bidirectional PIM and find solutions to these.

Bibliography

[ABG$^+$01] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, December 2001.

[ADF$^+$01] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas. LDP specification. Request for Comments 3036, Internet Engineering Task Force, January 2001.

[AGK99] George Apostolopoulos, Roch Guerin, and Sanjay Kamat. Implementation and performance measurements of qos routing extensions to OSPF. In *Proc. of IEEE INFOCOM'99*, pages 680–688, 1999.

[AJY00] C. Alaettinoglu, V. Jacobson, and H. Yu. Towards millisecond IGP congergence. Internet draft, draft-alaettinoglu-ISIS-convergence-00.ps, work in progress, November 2000.

[APN] APNIC. BGP Routing Table Analysis report. `http://bgp.potaroo.net/`.

[ATC$^+$04] A. Atlas, R. Torvi, G. Choudhury, C. Martin, B. Imhoff, and D. Fedyk. Ip/ldp local protection. Internet draft, draft-atlas-ip-local-protect-00.txt, work in progress, February 2004.

[Atl06] A. Atlas. U-turn alternates for IP/LDP fast reroute. Internet draft, draft-atlas-ip-local-protect-uturn-03.txt, work in progress, February 2006.

[AZ07] A. Atlas and A. Zinin. Basic Specification for IP Fast-Reroute: Loop-free Alternates. Internet draft, draft-ietf-rtgwg-ipfrr-spec-base-06, March 2007.

[BC96] T. Bates and R. Chandrasekeran. BGP route reflection an alternative to full mesh IBGP. Request for Comments 1966, Internet Engineering Task Force, June 1996.

[BEZ$^+$97] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation protocol (RSVP) – version 1 functional specification. Request for Comments 2205, Internet Engineering Task Force, September 1997.

[BFPS05] S. Bryant, C. Filsfils, S. Previdi, and M. Shand. IP Fast Reroute using tunnels. Internet draft, draft-bryant-ipfrr-tunnels-03.txt, work in progress, November 2005.

[BFSP06] O. Bonaventure, P. Francois, M. Shand, and S. Previdi. ISIS extensions for ordered FIB updates. Internet draft, draft-bonaventure-isis-ordered-00.txt, work in progress, February 2006.

[BLD$^+$07] S. Balon, J. Lepropre, O. Delcourt, F. Skivée, and G. Leduc. Traffic Engineering an Operational Network with the TOTEM Toolbox. In *IEEE Transactions on Network and Service Management*, volume 4, pages 51–61, June 2007.

[BLSZ03] R. Bless, G. Lichtwald, M. Schmidt, and M. Zitterbart. Fast Scoped Rerouting for BGP. In *International Conference on Networks*, pages 25–30. IEEE, September 2003.

[BRS03] Greg Bernstein, Bala Rajagopalan, and Debanjan Saha. *Optical Network Control: Architecture, Protocols, and Standards*. Addison-Wesley Professional, 2003.

[BS07] S. Bryant and M. Shand. A framework for loop-free convergence. Internet draft, draft-ietf-rtgwg-lf-conv-frmwk-01.txt, work in progress, June 2007.

[BSP06] S. Bryant, M. Shand, and S. Previdi. IP Fast Reroute Using Notvia Addresses. Internet draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-00.txt, work in progress, December 2006.

[Cai00] B. Cain. Fast link state flooding. In *GLOBECOM 2000 - IEEE Global Telecommunications Conference*, pages 465 – 469, November 2000.

[Cal90] R. W. Callon. Use of OSI IS-IS for routing in TCP/IP and dual environments. Request for Comments 1195, Internet Engineering Task Force, December 1990.

[Cas01] Stephen Casner. A fine-grained view of high-performance networking. Presented at NANOG22, http://www.nanog.org/mtg-0105/ppt/casner/sld001.htm, May 2001.

[CDZ97] K. Calvert, M. Doar, and E. Zegura. Modeling internet topology. *IEEE Communications Magazine 35:160–163*, 1997.

[CDZK05] J. Chandrashekar, Z. Duan, Z. Zhang, and J. Krasky. Limiting path exploration in BGP. In *IEEE INFOCOM*, Miami, Florida, March 2005.

[Cisa] Cisco Systems. IP Event Dampening. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s22/s_ipevdp.htm`.

[Cisb] Cisco Systems. IS-IS Fast-Flooding of LSPs Using the fast-flood Command. Available at `http://www.cisco.com/en/US/products/sw/iosswrel/ps1829/products_feature_guide09186a00801e87ab.html`.

[Cisc] Cisco Systems. SONET Triggers. `http://www.cisco.com/warp/public/127/sonetrig.pdf`.

[Cis03] Cisco Systems. IS-IS Support for Priority-Driven IP Prefix RIB Installation. Technical document, `http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s26/fslocrib.pdf`, 2003.

[Cis04a] Cisco. Prefix and Tunnel Independent FRR. `http://www.cisco.com/en/US/products/ps5763/prod_release_note09186a008033575a.html#wp98916`, November 2004.

[Cis04b] Cisco Systems. Configuring Integrated IS-IS. Technical document, `http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fipr_c/ipcprt2/1cfisis.htm`, 2004.

[CP04] S. De Cnodder and C. Pelsser. Protection for inter-AS MPLS tunnels, July 2004. Work in progress, draft-decnodder-ccamp-interas-protection-00.txt.

[DFM04] N. Dubois, B. Fondeviole, and N. Michel. Fast convergence project. Presented at RIPE47, http://www.ripe.net/ripe/meetings/ripe-47/presentations/ripe47-routing-fcp.pdf, January 2004.

[Dij59] Edsger. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[dSB07] V. Van den Schrieck and O. Bonaventure. Automating the iBGP organization in large IP networks. submitted to the ACM SIGCOMM Workshop on Internet Network Management 2007, May 2007.

[FABK03] N. Feamster, D. Andersen, H. Balakrishnan, and M. Kaashoek. Measuring the effects of internet path faults on reactive routing. In *ACM SIGMETRICS*, San Diego, CA (USA), June 2003.

[FB05] P. Francois and O. Bonaventure. Avoiding transient loops during IGP Convergence in IP Networks. In *Proc. IEEE INFOCOM*, March 2005.

[FBS$^+$06] P. Francois, O. Bonaventure, M. Shand, S. Bryant, and S. Previdi. Loop-free convergence using ordered FIB updates. Internet draft, draft-ietf-rtgwg-ordered-fib-00.txt, work in progress, December 2006.

[FE05] C. Filsfils and J. Evans. Deploying diffserv in IP/MPLS backbone networks for Tight SLA control. *IEEE Internet Computing*, 9(1):58–65, 2005.

[FFEB05] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure. Achieving subsecond IGP convergence in large IP networks. *SIGCOMM Comput. Commun. Rev.*, 35(3):35–44, 2005.

[FGL$^+$00] Anja Feldmann, Albert Greenberg, Carsten Lund, Nick Reingold, and Jennifer Rexford. Netscope: Traffic engineering for ip networks. *IEEE Network Magazine*, March 2000.

[Fil04a] C. Filsfils. Fast IGP convergence. Presented at RIPE47, http://www.ripe.net/ripe/meetings/ripe-47/presentations/ripe47-routing-isis.pdf, January 2004.

[Fil04b] C. Filsfils. IGP and BGP fast convergence. Networkers'2004, Cannes, France, December 2004.

[Fil05] C. Filsfils. IGP and BGP fast convergence. Tutorial presented at MPLS World Congress 2005, Paris (France), February 2005.

[Fil06] Clarence Filsfils. Bgp convergence in much less than a second. In *presented at NANOG40*, Bellevue, WA, US, June 2006. available from `http://www.nanog.org/mtg-0706/Presentations/ClarenceFilsfils-BGP.pdf`.

[FLH$^+$00] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. Generic routing encapsulation (GRE). Internet RFC 2784, March 2000.

[FMM$^+$04] A. Feldmann, O. Maennel, M. Mao, A. Berger, and B. Maggs. Locating internet routing instabilities. In *ACM SIGCOMM2004*, August 2004.

[FMR04] N. Feamster, Z. Mao, and J. Rexford. BorderGuard: Detecting Cold Potatoes from Peers. In *ACM Internet Measurement Conference*, Taormina, Italy, October 2004.

[FRT02] B. Fortz, J. Rexford, and M. Thorup. Traffic engineering with traditional IP routing protocols. *IEEE Communications Magazine*, October 2002.

[GEA] GEANT Network. `http://www.geant.net`.

[Gil05] V. Gill. AOL Transit Data Network AS1668 Building a Modern Backbone. Presentation at NANOG, `http://www.nanog.org/mtg-0505/`, May 2005.

[GLA89] J. J. Garcia-Luna-Aceves. A unified approach to loop-free routing using distance vectors or link states. *SIGCOMM Comput. Commun. Rev.*, 19(4):212–223, 1989.

[GM03] Vijay Gill and Jon Mitchell. AOL Backbone OSPF-ISIS Migration. NANOG, http://www.nanog.org/mtg-0310/gill.html, October 2003.

[GMG$^+$04] K. Gummardi, H. Madhyastha, S. Gribble, H. Leby, and D. Wetherall. Improving the reliability of Internet paths with One-hop Source Routing. In *USENIX OSDI'04*, 2004.

[GP01] T. Griffin and B. Presmore. An experimental analysis of BGP convergence time. In *ICNP 2001*, pages 53–61. IEEE Computer Society, November 2001.

[GR00] L. Gao and J. Rexford. Stable internet routing without global coordination. In *SIGMETRICS*, 2000.

[Gro05] S. Gross. Modern L2 VPNs : Implementing network convergence. Presentation at NANOG33, Feb 2005.

[GS02] Barry Raveendran Greene and Philip Smith. *Cisco ISP Essentials*. Cisco Press, 2002.

[GW02a] T. Griffin and G. Wilfong. Analysis of the MED oscillation problem in BGP. In *ICNP2002*, 2002.

[GW02b] T. Griffin and G. Wilfong. On the Correctness of IBGP Configuration. In *Proc. of ACM SIGCOMM*, 2002.

[Hed88] C. L. Hedrick. Routing information protocol. Request for Comments 1058, Internet Engineering Task Force, June 1988.

[HKM$^+$90] J. C. Honig, D. Katz, M. Mathis, Y. Rekhter, and J. Y. Yu. Application of the border gateway protocol in the internet. Request for Comments 1164, Internet Engineering Task Force, June 1990.

[HMMD02] U. Hengartner, S. Moon, R. Mortier, and C. Diot. Detection and analysis of routing loops in packet traces. In *Proceedings of the second ACM SIGCOMM Workshop on Internet measurment*, pages 107–112. ACM Press, 2002.

[ICBD04] G. Iannaccone, C. Chuah, S. Bhattacharyya, and C. Diot. Feasibility of IP restoration in a tier-1 backbone. *IEEE Network Magazine*, January-February 2004.

[ICM$^+$02] G. Iannaccone, C-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot. Analysis of link failures over an IP backbone. In *ACM SIGCOMM Internet Measurement Workshop*, Marseilles, France, November 2002.

[IIOY] Hiro Ito, Kazuo Iwama, Yasuo Okabe, and Takuya Yoshihiro. Avoiding Routing Loops on the Internet. In *Theory of Computing Systems*, volume 36, November.

[ISO02] ISO. Intermediate system to intermediate system routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (iso 8473). Tech. Rep. 10589:2002, ISO/IEC, April 2002.

[Jac] Dirk W. Jacob. SSF Implementation of OSPFv2 v0.2.2. `http://ssfnet.d-jacob.net/`.

[JM82] J. M. Jaffe and F. H. Moss. A Responsive Distributed Routing Algorithm for Computer Networks. *IEEE Trans. Commun.*, pages 30:1758–1762, July 1982.

[KCIB04] R. Keralapura, C. N. Chuah, G. Iannaccone, and S. Bhattacharyya. Service Availability: A New Approach to Characterize IP Backbone Topologies. In *Proceedings of IEEE IWQoS*, 2004.

[KW03] D. Katz and D. Ward. BFD for IPv4 and IPv6 (Single Hop). Internet draft, draft-katz-ward-bfd-v4v6-1hop-00.txt, work in progress, August 2003.

[KW06] D. Katz and D. Ward. Bidirectional Forwarding Detection. Internet draft, draft-ietf-bfd-base-05.txt, work in progress, June 2006.

[LABJ00] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. An experimental study of internet routing convergence. In *SIGCOMM 2000*, August 2000.

[LTG04] J. Lau, M. Townsley, and I. Goyret. Layer two tunneling protocol - version 3 (L2TPv3). Internet draft, draft-ietf-l2tpext-l2tp-base-15.txt, work in progress, December 2004.

[LYN$^+$04] S. Lee, Y. Yu, S. Nelakuditi, Z.-L. Zhang, and C.-N. Chuah. Proactive vs. reactive approaches to failure resilient routing. In *Proc. IEEE INFOCOM"*, March 2004.

[MAMB01] A. Medina, A.Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *MASCOTS 2001*, August 2001.

[Mar02] Abe Martey. *IS-IS Network Design Solutions*. Cisco Press, 2002.

[MIB$^+$04] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and Ch. Diot. Characterization of failures in an IP backbone. In *IEEE Infocom2004*, Hong Kong, March 2004.

[Moy91] J. Moy. OSPF version 2. Request for Comments 1247, Internet Engineering Task Force, July 1991.

[MPEL03] J. Moy, P. Pillay-Esnault, and A. Lindem. Graceful OSPF Restart. Internet RFC 3623, November 2003.

[MRR79] John M. McQuillan, Ira Richer, and Eric C. Rosen. An overview of the new routing algorithm for the arpanet. In *Proceedings of the sixth symposium on Data communications*, pages 63–68. ACM Press, 1979.

[Nai04a] V. Naidu. Ip fast reroute using multiple path algorithm (mpa). Internet draft, draft-venkata-ipfrr-mpa-00.txt, work in progress, June 2004.

[Nai04b] V. Naidu. Ip fast reroute using stitched shortest paths algorithm (sspa). Internet draft, draft-venkata-ipfrr-sspa-00.txt, work in progress, June 2004.

[ND05] B. Fondeviole N. Dubois, B. Decraene. "Requirements for planned maintenance of BGP sessions". Internet draft, draft-dubois-bgp-pm-reqs-00.txt, work in progress, February 2005.

[NLYZ03] S. Nelakuditi, S. Lee, Y. Yu, and Z. Zhang. Failure Insensitive Routing for Ensuring Service Availability. In *IWQoS*, 2003.

[NST99] P. Narvez, K. Siu, and H. Tzeng. Local Restoration Algorithms for Link-State Routing Protocols. In *ICCC'1999*, Boston, Massachussetts, October 1999. available from `http://perth.mit.edu/~pnarvaez/publications.html`.

[Ora90] D. Oran. OSI IS-IS intra-domain routing protocol. Request for Comments 1142, Internet Engineering Task Force, February 1990.

[PAN$^+$05] D. Pei, M. Azuma, N. Nguyen, J. Chen, D. Massey, and L. Zhang. BGP-RCN: Improving BGP convergence through Root Cause Notification. *Computer Networks*, 48(2):175–194, June 2005. 2005.

[PDRG02] P. Pongpaibool, R. Doverspike, M. Roughan, and J. Gottlieb. Handling ip traffic surges via optical layer reconfiguration. *Optical Fiber Communication*, 2002.

[PLM$^+$03] D. Pei, L.Wang, D. Massey, S. F. Wu, and L. Zhang. A Study of Packet Delivery Performance During Routing Convergence. In *Proceedings of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2003.

[PMA01] J. Parker, D. McPherson, and C. Alaettinoglu. Short Adjacency Hold Times in IS-IS. Internet draft, draft-parker-short-isis-hold-times-01.txt, work in progress, July 2001.

[QB02] B. Quoitin and O. Bonaventure. A survey of the utilization of the BGP community attribute. Internet draft, draft-quoitin-bgp-comm-survey-00.txt, work in progress, March 2002.

[QUP$^+$03] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure. Interdomain traffic engineering with BGP. *IEEE Communications Magazine*, May 2003.

[Rei04] C. Reichert. IP-protection for fast inter-domain resilience. Presented at IDRWS'04, available from `http://www.tm.uka.de/idrws/2004/contributions/IDRWS2004--08--Reichert_Christoph--IP_Protection_for_Fast_InterDomain_Resilience.pdf`, May 2004.

[Rek91] Yakov Rekhter. Constructing intra-AS path segments for an inter-AS path. *SIGCOMM Comput. Commun. Rev.*, 21(1):44–57, 1991.

[Ren] Renesys. SSFNet, Scalable Simulation Framework for Network Models. `http://www.ssfnet.org/`.

[RG94] Y. Rekhter and P. Gross. Application of the border gateway protocol in the internet. Request for Comments 1655, Internet Engineering Task Force, July 1994.

[RLH06] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). Request for Comments 4271, Internet Engineering Task Force, 2006.

[Ros04] Carsten Rossenhovel. 40-gig router test results. *Light Reading*, November 2004. Available from `http:`

`//www.lightreading.com/document.asp?site=testing&doc_id=63606&page_number=6`.

[RPAW05] R. Raszuk, K. Patel, C. Appanna, and D. Ward. BGP aggregate withdraw. Internet draft, draft-raszuk-aggr-withdraw-00.txt, work in progress, October 2005.

[RR99] E. Rosen and Y. Rekhter. BGP/MPLS VPNs. Request for Comments 2547, Internet Engineering Task Force, March 1999.

[RR01] Y. Rekhter and E. Rosen. Carrying label information in BGP-4. Request for Comments 3107, Internet Engineering Task Force, May 2001.

[RR04] Y. Rekhter and E. Rosen. Use of PE-PE GRE or IP in BGP/MPLS IP Virtual Private Networks. Internet draft, draft-ietf-l3vpn-gre-ip-2547-03.txt, work in progress, October 2004.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. Request for Comments 3031, Internet Engineering Task Force, January 2001.

[RWS00] A. Retana, R. White, and D. Slice. *EIGRP for IP: Basic Operation and Configuration*. Addison Wesley, 2000.

[SB07] M. Shand and S. Bryant. IP Fast Reroute Framework. March 2007.

[SCK$^+$03] G. Schollmeier, J. Charzinski, A. Kirstädter, C. Reichert, K. Schrodi, Y. Glickman, and C. Winkler. Improving the Resilience in IP Networks. In *High performance switching and routing (HPSR'03)*, Torino, June 2003.

[SDV02] A. Shaikh, R. Dube, and A. Varma. Avoiding Instability during Graceful Shutdown of OSPF. In *Proc. IEEE INFOCOM*, June 2002.

[SG01] Aman Shaikh and Albert Greenberg. Experience in black-box OSPF measurement. In *Proceedings of the First ACM SIGCOMM Workshop on Internet Measurement*, pages 113–125. ACM Press, 2001.

[SG04a] A. Shaikh and A. Greenberg. OSPF Monitoring: Architecture, Design and Deployment Experience. In *Proc. USENIX Symposium on Networkeds System Design and Implementation (NSDI)*, March 2004.

[SG04b] M. Shand and L. Ginsberg. Restart signaling for IS-IS. Internet RFC 3847, July 2004.

[Sie02] D. Siegel. Operationnal experience with mpls. NANOG, http://www.nanog.org/mtg-0202/ppt/siegel/sld001.htm, February 2002.

[SIG$^+$02] Aman Shaikh, Chris Isett, Albert Greenberg, Matthew Roughan, and Joel Gottlieb. A case study of ospf behavior in a large enterprise network. In *Proceedings of the second ACM SIGCOMM Workshop on Internet measurment*, pages 217–230. ACM Press, 2002.

[SMW02] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *SIGCOMM 2002*, 2002.

[SP03] N. Shen and P. Pan. Nexthop Fast ReRoute for IP and MPLS. Internet draft, draft-shen-nhop-fastreroute-00.txt, work in progress, December 2003.

[SS04] N. Shen and H. Smit. Calculating Interior Gateway Protocol (IGP) Routes Over Traffic Engineering tunnels. Internet RFC 3906, October 2004.

[STR04] S. Sangli, D. Tappan, and Y. Rekhter. BGP extended communities attribute. Internet draft, draft-ietf-idr-bgp-ext-communities-07.txt, work in progress, April 2004.

[SVKD00] Aman Shaikh, Anujan Varma, Lampros Kalampoukas, and Rohit Dube. Routing stability in congested networks: experimentation and analysis. *SIGCOMM Comput. Commun. Rev.*, 30(4):163–174, 2000.

[Tec05] Cariden Technologies. Multiprotocol Automation and Traffic Engineering (MATE). http://www.cariden.com/, 2005.

[TMS01] P. Traina, D. McPherson, and J. Scudder. Autonomous system confederations for BGP. Request for Comments 3065, Internet Engineering Task Force, February 2001.

[TMSV03] R. Teixeira, K. Marzullo, S. Savage, and G.M. Voelker. In search of path diversity in ISP networks. In *USENIX/ACM Internet Measurement Conference*, October 2003.

[TR06] Renata Teixeira and Jennifer Rexford. Managing routing disruptions in Internet Service Provider networks. *IEEE Communications Magazine*, March 2006.

[TSGR04] R. Texeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. In *SIGMETRICS/Performance'04*, New York, NY, USA, June 2004.

[Var04] George Varghese. *Network algorithmincs*. Morgan Kaufmann, 2004.

[VPD04] J.-P. Vasseur, M. Pickavet, and P. Demeester. *Network Recovery: Protection and Restoration of Optical, SONET-SDH, and MPLS*. Morgan Kaufmann, 2004.

[WJL03] David Watson, Farnam Jahanian, and Craig Labovitz. Experiences with monitoring ospf on a regional service provider network. In *Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 204. IEEE Computer Society, 2003.

[WMS04] R. White, D. McPherson, and S. Sangli. *Practical BGP*. Addison Wesley, 2004.

[WMW$^+$06] F. Wang, Z. Morley Mao, J. Wang, L. Gao, and R. Bush. A measurement study on the impact of routing events on end-to-end internet path performance. In *SIGCOMM'2006*, Pisa, Italy, September 2006.

[WR03] R. White and A. Retana. *IS-IS: Deployment in IP networks*. Addison-Wesley, 2003.

[WRC05] D. Walton, A. Retana, and E. Chen. "Advertisement of Multiple Paths in BGP". Internet draft, draft-walton-bgp-add-paths-04.txt, work in progress, August 2005.

[WRR04] T. Worster, Y. Rekhter, and E. Rosen. Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE). Internet draft, draft-ietf-mpls-in-ip-or-gre-08.txt, work in progress, June 2004.

[XMP04] X. Xiao, D. McPherson, and P. Pate. Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3). Request for Comments 3916, Internet Engineering Task Force, 2004.

[ZKN$^+$05] Z. Zhong, R. Keralapura, S. Nelakuditi, Y. Yu, J. Wang, C. Chuah, and S. Lee. Avoiding transient loops through interface-specific forwarding. In *IWQoS'05*, 2005.