UNIVERSITY OF CALIFORNIA, SAN DIEGO

Network Sensitivity to Intradomain Routing Changes

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in Computer Science and Engineering

by

Renata Teixeira

Committee in charge:

Professor Geoffrey M. Voelker, Chair
Professor Keith Marzullo
Professor Stefan Savage
Professor Rene Cruz
Professor Fan Chung-Graham
Professor Jennifer Rexford
Professor Timothy Griffin

2005

The dissertation of Renata Teixeira is approved, and it is acceptable in quality and form for publication on microfilm:

_____

_____

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2005

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The six years it took me to complete the Ph.D. program were extremely enriching because of the amazing people I had the chance to interact with. First, I would like to thank my advisor at UCSD, Prof. Geoff Voelker, and my mentors at AT&T, Prof. Jennifer Rexford and Prof. Tim Griffin. Geoff's (almost) unconditional support gave me the strength and courage I needed to go through every one of the many hurdles that necessarily arise during a Ph.D. Jennifer and Tim helped me frame the topic of my thesis while at AT&T Labs. Tim and our inspiring white-board discussions always forced me to question my beliefs and find the essence of every problem. On the surface, writing an acknowledgment for Jennifer should be easy as she worked very closely with me at every step of this thesis and guided me in many important decisions. However, it is hard to make justice to her role in a few sentences.

Before going to AT&T, I had three advisors: Geoff, Keith Marzullo, and Stefan Savage. Keith has been mentoring me since very early in my Ph.D. Stefan's "right-to-the-point" questions had great impact in my research direction. Only Geoff survived, but Keith and Stefan made me the honor of joining my committee. I also thank the other members of my Ph.D. committee, Prof. Fan Chung-Graham and Prof. Rene Cruz for their feedback, and Prof. George Varghese and Prof. Fran Berman for their encouragement.

I am grateful to AT&T, specially Albert Greenberg, for giving me access to measurement data of their network, which was the starting point of this work. I thank Aman Shaikh for his patience in helping me understand the network and its monitoring system. I also thank my other co-authors Mauricio Resende, Matt Roughan, and Nick Duffield for their contribution to this work. I thank Abilene for making their measurement data publicly available to the research community. I am grateful to Anukool Lakhina for his help in working with the Abilene data.

These years would not have been as much fun if it was not for the CSL-south

crew. I will miss my daily chats with my cube-mates Ranjita Bhagwan and Leeann Bent. John Bellardo and Marvin McNett's technical support have been essential for my peace of mind to finish this work. Flavio Junqueira was my study-buddy in the beginning of the program, and thank him for our discussions.

I am extremely grateful to my family for supporting me in the pursuit of my dreams, even when it meant to be away from them. Finally, I want to thank my husband, Christophe, for never letting me give up and always challenging me to overcome what I thought were my limits.

The text of the following chapters, in part or in full, are based on material that has been published in conference or workshop proceedings. Chapters 3 and 4 are based on material published in the ACM SIGMETRICS Conference, 2004 and the PAM Workshop, 2005. Chapter 5 is based on material published in the ACM SIGCOMM Conference, 2004. Chapter 6 is based on material presented in the AT&T technical report TD-69EJBE, 2005 and presented in the CoNEXT Conference, 2005. I was the primary researcher and author of each of the above publications, and the co-authors listed in these publications collaborated on, or supervised the research which forms the basis for these chapters.

VITA

1997	B.Sc., Computer Science, Universidade Federal do Rio de Janeiro
1999	M.Sc., Eletrical Engineering COPPE/Universidade Federal do Rio de Janeiro
1999–2005	Graduate Teaching and Research Assistant, Dept. of Computer Science and Engineering, University of California, San Diego
2003,2004	Summer Intern, AT&T Labs – Research
2005	Ph.D., Computer Science and Engineering, University of California, San Diego

PUBLICATIONS

R. Teixeira, T. Griffin, M.G.C. Resende, and J. Rexford, " TIE Breaking: Tunable Interdomain Egress Selection", in *CoNEXT*, October 2005.

R. Teixeira, N. Duffield, J. Rexford, and M. Roughan, " Traffic Matrix Reloaded: Impact of Routing Changes", in *Passive and Active Measurement Workshop*, April 2005.

R. Teixeira, T. Griffin, A. Shaikh, and G.M. Voelker, "Network Sensitivity to Hot-Potato Disruptions", in *Proc. ACM SIGCOMM*, Portland, Oregon, September 2004.

R. Teixeira and J. Rexford, "A Measurement Framework for Pin-Pointing Routing Changes", in *Proc. ACM SIGCOMM Network Troubleshooting Workshop*, Portland, Oregon, September 2004.

R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford, "Dynamics of Hot-Potato Routing in IP Networks", in *Proc. ACM SIGMETRICS*, New York City, New York, June 2004.

R. Teixeira, K. Marzullo, S. Savage, and G.M. Voelker, "In Search of Path Diversity in ISP Networks", in *Proc. USENIX/ACM Internet Measurement Conference*, Miami, Florida, October 2003.

R. Teixeira, K. Marzullo, S. Savage, and G.M. Voelker, "Characterizing and Measuring Path Diversity in Internet Topologies", in *Proc. ACM SIGMETRICS* (extended abstract), San Diego, California, June 2003.

ABSTRACT OF THE DISSERTATION

Network Sensitivity to Intradomain Routing Changes

by

Renata Teixeira
Doctor of Philosophy in Computer Science and Engineering
University of California, San Diego, 2005
Professor Geoffrey M. Voelker, Chair

The Internet's routing architecture was designed to have a clean separation between the intradomain and interdomain routing protocols. However, the appropriate "division of labor" between these two tiers becomes unclear when an Autonomous System (AS) has interdomain routes to a destination through multiple border routers – a situation that is extremely common today because neighboring domains often connect in several locations. Unfortunately, this evolution in Internet structure has made it increasingly susceptible to unforeseen interactions between the two routing protocols. We believe that the current mechanism of *early-exit* or *hot-potato* routing—where each router in an AS directs traffic to the "closest" border router based on the intradomain distances—is convoluted, restrictive, and sometimes quite disruptive.

This thesis improves the robustness of IP networks by revisiting the interaction between intradomain and interdomain routing protocols. First, it analyzes the influence of intradomain routing changes on BGP routing (the interdomain routing in the Internet today). We found that some intradomain routing changes trigger a significant number of BGP updates. In fact, these BGP routing changes are responsible for the largest traffic variations. Applications such as voice over IP, streaming, and gaming are particularly sensitive to these instabilities.

As a result, the development of guidelines and tools for the design and configuration of networks that minimize the impact on BGP are important tasks for achieving network robustness. We address these challenges using an analytic model of routing interaction that incorporates metrics to evaluate network sensitivity to intradomain changes. Our model identifies vulnerabilities in the network and can be used by network administrators to engineer more robust networks.

Finally, we propose a simple change to router's BGP decision logic to implement a flexible mechanism for selecting egress points for traffic. This mechanism allows network administrators to satisfy diverse goals, such as traffic engineering and robustness to equipment failures. We present two example optimization problems that use integer-programming and multicommodity-flow techniques, respectively, to tune our mechanism to satisfy network-wide objectives. Experiments with traffic, topology, and routing data from two backbone networks demonstrate that our solution is both simple (for the routers) and expressive (for the network administrators).

Chapter 1

Introduction

The Internet has become a critical part of today's communication infrastructure. This success brings increasing pressure for Internet Service Providers (ISPs) to provide the quality of service needed by a wide range of applications. Applications such as voice over IP, video games, and commercial transactions cannot tolerate neither persistent performance problems nor transient disruptions. In particular, these applications are sensitive to high delay, delay variation, low available bandwidth, and loss.

Unfortunately, routine events such as network failures and maintenance trigger routing changes that cause service disruptions. For example, a study of the performance of voice over IP (VoIP) found that most service disruptions are not related to delay or congestion, but happen during network failures [BID02]. In addition to the transient packet losses and delays during routing convergence, shifting traffic to new routes may cause congestion or changes in other path properties (e.g., a higher round-trip time). Even a few hundred milliseconds of routing instability is long enough to interrupt a phone conversation and a video game, and other applications such as Web transactions are visibly affected by disruptions lasting a few seconds. Longer periods of congestion or long delay make it impossible to use these services.

Providing quality of service on today's routing system is a challenging task: there is not enough flexibility of control to minimize transient disruptions or to control

the flow of traffic. In this thesis, we analyze measurements of routing and traffic from a large IP backbone and identify that the boundary between intradomain and interdomain routing is an important source of routing instability. In particular, interdomain routing instabilities arise because of the current mechanism for selecting egress routers in an IP network. We then propose a model that captures the impact of events that happen inside an IP network on interdomain routing and traffic. Network administrators can use this model to minimize transient disruptions. However, we argue that the problem of selecting egress routers deserves further analysis. We study this problem in more detail, and propose a new mechanism for selecting egress routers to replace the current one. Our mechanism allows ISPs to provide quality of service by giving them finer control of the flow of traffic in transient or steady state. As a result, ISPs can attend the performance requirements of a wide range of applications.

1.1 Operational View of IP Routing

ISPs seek to build *robust* networks so that small perturbations in the environment or internal conditions do not significantly impact network performance. Network administrators in ISPs juggle current technologies to provide good steady-state performance and to avoid or reduce transient performance disruptions to their customers' traffic as much as possible. Maintenance of network equipment is usually done in periods of lower utilization to minimize disruption. Most large ISP backbone networks are overprovisioned to avoid losses due to congestion, and network administrators use tools to predict the impact of network changes on link utilization. To reduce performance degradation during routing convergence, ISPs work with router vendors to reduce convergence delays and employ a graceful shutdown of links and routers that need to undergo maintenance. ISPs also deploy monitoring infrastructure to help pinpoint performance problems as soon as they manifest themselves.

Despite all this effort, in practice robustness is not so easily achieved. In fact,

robustness is not even an easy thing to measure. A network can be running smoothly only to crash unexpectedly after some seemingly "small" event. A robust network should have low sensitivity to routing changes and traffic load variations. To understand the robustness of an IP network, it is important to understand the robustness of both the *control plane*, which relates to the behavior of routing protocols, and *data plane*, which relates to packet forwarding. For example, a small routing change inside a network may have significant impact in the control plane by causing an overwhelming number of interdomain routing messages — which in turn may overload and crash routers, even though the routes initially in play were carrying no traffic. On the other hand, small routing changes on paths to popular destinations can impact the data plane by causing a large swing in traffic, perhaps leading to congestion, losses, delay, and jitter.

The delivery of IP traffic depends on the routing protocols running in and among thousands of Autonomous Systems (ASes). The Internet's two-tiered routing system allows the separation of intradomain and interdomain routing, each emphasizing different design goals. Interior Gateway Protocols (IGPs) are typically "metric based" to give administrators control over resource allocation within their networks. For example, OSPF [Moy98] and IS-IS [Cal90] compute shortest paths based on link weights assigned by administrators. In contrast, the design of the Border Gateway Protocol (BGP) [RLH04] emphasizes scalability to a large number ASes and destinations, as well as the use of locally-configurable routing policies. Routing policies are important to express business relationships between ASes (for instance, a customer AS should learn routes to all other destinations in the Internet from its provider AS). BGP is responsible for exchanging route advertisements with neighboring domains and propagating reachability information within an AS. A router combines the BGP and IGP information to determine the forwarding path to the destination. Although IGP weights and BGP policies are supposed to be static, network administrators often need to reconfigure them for doing traffic engineering. In fact, these are the primary mechanisms that network admin-

istrators have available to control the flow of traffic. They reconfigure IGP weights when there is internal congestion [FRT02, NSTD03] and BGP policies to alleviate congestion at peering links [FBR03, Uhl04].

The two-tiered routing architecture should isolate the global Internet from routing changes within an individual AS. However, in practice, the interaction between intradomain and interdomain routing is more complicated. Despite the distinct functionality, the intradomain and interdomain routing protocols need to cooperate to ensure end-to-end packet delivery. For example, Figure 1.1 shows an AS with two BGP sessions with a neighboring AS that advertises routes to a destination d. The two routers A and B propagate this information to router C. For routers inside the AS both routes to reach the outside destination d (learned from A and B) look "equally good" at the BGP level (with the same local preference, number of AS hops, etc.). This leaves C with the dilemma of choosing between egress points A and B to forward packets to d. Under the common practice of *early-exit* or *hot-potato* routing, C directs traffic to the *closest* egress point—the router with the smallest intradomain distance (i.e., router A with distance 9 from C). The intent of hot-potato routing is to limit the bandwidth resources consumed by the traffic by moving packets to a neighboring AS at the nearest opportunity.

However, suppose the IGP distance to A changes from 9 to 11, in response to a link failure along the original path or an intentional change in a link weight for traffic engineering or planned maintenance. Although the BGP route through A is still available, the IGP distance change would cause C to select the route through egress point B. We refer to this as a *hot-potato routing change*. Hot-potato routing changes can have a significant performance impact on applications due to: (i) transient packet delay and loss while the routers recompute their forwarding tables, (ii) shifts in traffic that may cause congestion on the new paths through the network, and (iii) BGP routing changes visible to neighboring domains. The frequency and importance of these effects

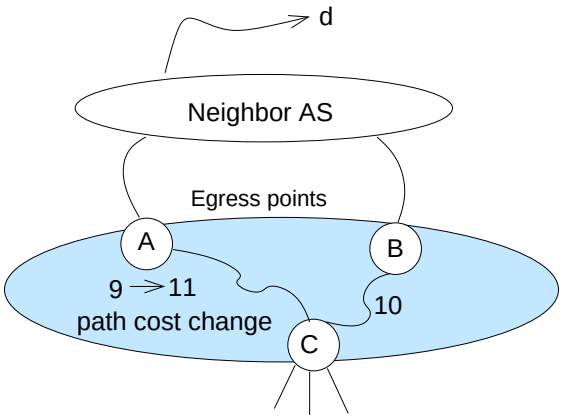


Figure 1.1: Example of interaction between IGP and BGP to select the route to an external destination d.

depend on a variety of factors. A large ISP network connects to many neighboring domains in many geographic locations. In practice, an ISP typically learns "equally good" BGP routes at each peering point with a neighboring AS, which increases the likelihood that routing decisions depend on the IGP distance to the egress points. In addition, the routers have BGP routes for more than 100,000 blocks of destinations, and a single IGP distance change may cause many of these routes to change at the same time. If these destinations receive a large volume of traffic, the influence on the flow of traffic within the AS and on its downstream neighbors can be quite significant. One contribution of this thesis is to quantify these effects by analyzing the IGP-triggered BGP updates in a large ISP backbone network.

1.2 The Problem of Egress-Point Selection

The appropriate role of the routing protocols is unclear when the AS learns routes to a destination from more than one egress router. Hot-potato routing represents the policy of selecting the closest egress router. Under this policy the IGP weights have overloaded semantics. They are used to determine both *intradomain paths* and *each*

router's selection of egress point. These are two distinct functions, and we demonstrate that they should be decoupled.

Having multiple egress routers that reach an external destination is extremely common in the contemporary Internet. Since ISPs peer in multiple locations, essentially *all* of the traffic from customers to the rest of the Internet has multiple egress routers. In addition, many customers connect to their provider in multiple locations for fault tolerance and more flexible load balancing, resulting in multiple egress routers for these destinations as well. The multiple connections between ASes is now a fundamental part of the Internet's routing architecture, independent of the current set of intradomain and interdomain routing protocols. Therefore, there are three distinct parts of the routing system: (i) interdomain routing, which determines the set of border (or *egress*) routers that direct traffic toward a destination, (ii) intradomain routing, which determines the path from an ingress router to an egress router, and (iii) egress-point selection, which determines which egress router each ingress router choses for each destination. Although these are three equally-important aspects of the routing system, great attention has been given to the first two, but not to the third. Therefore, this thesis evaluates hot-potato routing in today's networks, and designs more sophisticated techniques for selecting egress points.

The limitation of hot-potato routing is that it confounds internal and external metrics, which forces administrators to consider the impact on interdomain routing when they make changes to internal network design. Egress-point selection and intradomain paths are two very distinct functions. Paths between two routers inside the network should be selected to minimize some meaningful performance metric, whereas the egress-point selection should be flexible to allow a wider range of traffic engineering goals. One contribution of this thesis is to demonstrate that it is feasible to decouple the two tiers of the routing system, and that this separation leads to greater stability and flexibility.

1.3 Contributions

In this thesis, we address the problem of egress-point selection. First, we evaluate the impact of the current egress-point selection mechanism (hot-potato routing) in an operational network. Then, we model this mechanism and introduce metrics of network sensitivity to internal changes that network administrators can use to engineer more robust networks. Finally, we propose a flexible mechanism for replacing hot-potato routing that allows the implementation of a wide variety of egress selection policies. The main contributions of this thesis are as follows.

1.3.1 Measurement and Characterization of the Impact of Hot-Potato Routing

To characterize the influence of hot-potato routing on IP networks, we propose a technique for associating BGP routing changes with events visible in the intradomain protocol. Our algorithm for correlating IGP and BGP data (i) generates a sequence of distance changes that may affect BGP routing decisions, (ii) classifies BGP routing changes in terms of possible IGP causes, and (iii) matches BGP routing changes with related IGP distance changes that occur close in time. Then, we apply our algorithm to routing data collected from a large ISP network, and identify suitable values for the parameters of the algorithm.

We show that (i) hot-potato routing can be a significant source of BGP updates and can be responsible for the largest traffic shifts, (ii) BGP updates can lag sixty seconds or more behind the intradomain event, (iii) the number of BGP routing changes triggered by hot-potato routing has a nearly uniform distribution across destinations, and (iv) the fraction of BGP messages triggered by intradomain changes varies significantly across time and router locations. We show that hot-potato routing changes lead to longer delays in routing convergence, shifts in the flow of traffic to neighboring domains, extra externally-visible BGP update messages, and inaccuracies in Internet performance measurements.

1.3.2 Model of Network Sensitivity to Hot-Potato Disruptions

Our measurement study shows that hot-potato routing can have a substantial impact on large ISP backbones and thereby jeopardize network robustness. As a result, there is a need for guidelines and tools to assist in the design of networks that minimize hot-potato disruptions. However, developing these tools is challenging due to the complex and subtle nature of the interactions between exterior and interior routing. To address these challenges, we introduce an analytic model of hot-potato routing that incorporates metrics to evaluate network sensitivity to hot-potato disruptions. We then present a methodology for computing these metrics using measurements of real ISP networks.

We demonstrate the utility of our model by analyzing the sensitivity of a large AS in an ISP network. Our case study of the control plane sensitivity shows that, on average, the ISP network is very robust to link and router failures. Nevertheless, there is room for improvement: our study pointed out some link and router failures that can cause routers to shift egress points for a large number of destinations. Network administrators can use this model to identify vulnerabilities in the network, and engineer it to be more robust to internal changes. We describe how certain design guidelines and operational practices can help prevent unnecessary hot-potato routing changes.

1.3.3 A Mechanism for Egress-Point Selection

Our model assists network administrators in engineering their networks assuming routing protocols as they are today. However, network management and traffic engineering could be greatly simplified if there were a more flexible mechanism for selecting egress points. Hot-potato routing mixes policy and mechanism: the policy of selecting the closest egress point is hard-wired in the BGP decision process at each router. We believe that this mechanism is convoluted, restrictive, and sometimes quite disruptive. In this thesis, we propose a flexible mechanism for routers to select the egress

point for each destination prefix called Tunable Interdomain Egress selection (TIE). The TIE mechanism we propose is: (i) flexible in balancing the trade-off between sensitivity to IGP changes and adaptability to network events, (ii) computationally easy for the routers to execute in real time, and (iii) easy for a higher-level management system to optimize based on diverse network objectives.

TIE allows network administrators to satisfy diverse goals, such as traffic engineering and robustness to equipment failures. We present example problems that can be solved easily using TIE that cannot be done today. First, we show how to minimize sensitivity to internal topology changes, subject to a bound on propagation delay, using integer programming to tune the weights in our mechanism. Second, we show how to balance load in the network without changing the IGP metrics or BGP policies by using multicommodity-flow techniques to move some traffic to different egress points. We evaluate the effectiveness of TIE for the two optimization problems, using traffic, topology, and routing data from two backbone networks (Abilene and a large tier-1 ISP). Our experiments show that TIE reduces sensitivity to internal topology changes while satisfying network-wide objectives for load and delay.

1.4 Overview of the dissertation

This dissertation is organized as follows. Chapter 2 presents background on Internet routing and traffic engineering. Chapter 3 introduces a methodology for measuring the importance of hot-potato routing changes in operational networks. Then, we apply this methodology to the backbone network of a large ISP and characterize the impact of hot-potato routing in Chapter 4. Chapter 5 presents our analytic model of network sensitivity to internal routing changes assuming hot-potato routing as the egress selection policy. Chapter 6 introduces TIE, a more flexible mechanism for egress point selection, and defines a range of policies that express different traffic engineering goals. We conclude in Chapter 7.

Chapter 2

Internet Routing Architecture

This chapter presents basic concepts of Internet routing as background to this thesis. First, we present a brief overview of the Internet as a network of networks. Then, we discuss each tier of the routing system in isolation, and the interaction between the two tiers. Throughout this discussion we stress the issues that have been subject of study by the research or the network operations communities.

2.1 Overview of Internet Routing

The Internet is an interconnection of Autonomous System's (AS) networks, which are administered by Internet Service Providers (ISPs) and their customers. Providers sell Internet connectivity to customers, which are usually universities, enterprises, or other providers. An ISP may need to contract another provider to connect to all networks that belong to the Internet. An ISP that does not have any other provider is called a *tier-1* ISP.Figure 2.1 presents a simple internetwork with five ASes. Say that both Big ISP and Large ISP are tier-1 ISPs. In this example, Big and Large *peer* in three locations. Small and Medium ISPs are customers of both Big and Large. Because Small and Medium have more than one provider, they are called *multi-homed* customers. The campus network UnivNet is a customer of Medium ISP only.

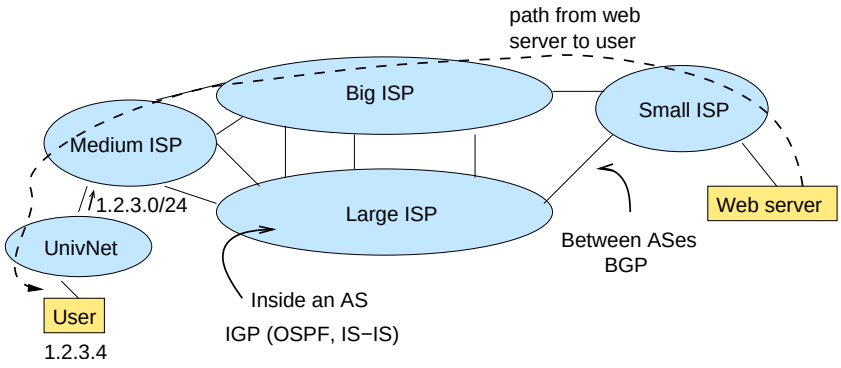


Figure 2.1: Example of an internetwork.

For the user to be able to download content from the web server, all ASes in the path between the two need to cooperate (as indicated by the dotted arrow). Routers in all ISP networks need to learn how to reach the IP address of the user. In the Internet, the Border Gateway Protocol (BGP) [RL95, RLH04] is the interdomain routing protocol responsible for exchanging reachability information of external *destination prefixes*, or blocks of IP addresses that belong to other ASes. In the example in Figure 2.1, the IP address of the user machine is $1.2.3.4$, which belongs to the IP address prefix $1.2.3.0/24$ allocated to UnivNet. BGP is responsible for selecting the *AS path* (or which ASes have to be traversed) to reach a destination prefix. Inside each network the intradomain routing protocol, or Interior Gateway Protocol (IGP), is responsible for selecting the path between the ingress router and the egress router.

This distributed management of the Internet by different domains was an important design goal from the early days of the ARPANET [Cla88]. The Internet's two-tiered routing system allows ASes to exchange routing information without divulging their internal details to each other. This division also allows each AS to select its own intradomain routing protocol. Although this architecture clearly separates the functions of intradomain from interdomain routing, at each domain BGP and IGP do have to interact to assure end-to-end packet delivery. In fact, the vast majority of the traffic in the

Internet today traverses multiple ASes, and consequently is routed using both protocols.

We illustrate the interaction between the two tiers of the routing system in Figure 2.2. This figure takes a closer look at the network of Small ISP from Figure 2.1. The traffic from the web server to the user enters the Small ISP network at ingress router A and has two choices of egress routers: B or C. BGP is responsible for selecting which egress router to use to forward traffic to the user; say that it selects egress C. Then, the IGP is responsible for selecting the path from A to C. In this example, router A combines the BGP and IGP information to construct a forwarding table that maps destination prefixes to outgoing links. BGP selects egress router C for the user's prefix, and IGP determines that x is the next hop in the path to C. Then, A installs an entry in its forwarding table mapping the user's prefix to x.

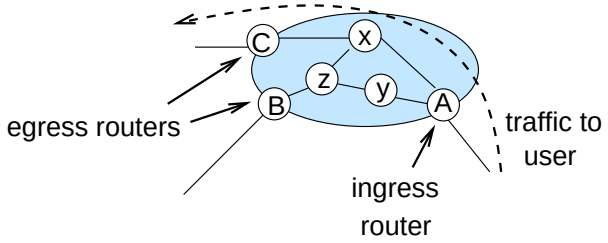


Figure 2.2: Example of the internal network topology of the Small ISP in Figure 2.1.

This section presents a high-level overview of the Internet routing architecture. We have summarized the roles of intradomain and interdomain routing, and shown that these two tiers need to interact to provide end-to-end packet delivery. The rest of this chapter describes each part of the routing architecture in more detail.

2.2 Intradomain routing

Intradomain routing protocols compute shortest paths based on a "metric" defined by network administrators. The most common routing protocols in large ISPs today are OSPF [Moy98] and IS-IS [Cal90]. These are link-state routing protocols where

each unidirectional link is assigned an administrative weight. The default value of IGP link weights is the inverse of the capacity of the link [Cise]. However, network administrators often configure link weights to achieve some traffic engineering goal such as balancing the traffic load in the network [FRT02, NSTD03].

The reliable flooding of link-state messages ensures that each router can construct a complete view of the network topology. IGP messages are flooded periodically and in response to topology changes, such as link weight changes and equipment going up or down. Link weights only change when reconfigured by network administrators, not dynamically to adapt to changes in network conditions such as congestion. Each router runs Dijkstra's algorithm [Dij59] to compute the shortest paths to every other node and uses the results to build the forwarding table. This ensures that, in steady state, each IP packet is forwarded along a shortest path in terms of link weights. For scalability, both OSPF and IS-IS allows the network to be divided into areas to define a two-level hierarchy. Area 0 (the backbone area) resides at the top level of the hierarchy and provides connectivity to the other areas. When areas are used the details of the topology inside each area are hidden from the other areas.

The behavior of intradomain routing protocols has a direct impact on end-user performance. IGP paths determine the experienced delay and available bandwidth, and intradomain routing anomalies can cause longer delays, packet re-ordering, and packet loss. Both the research and network operations communities have dedicated great effort in: understanding IGP behavior by monitoring routing inside an AS; reducing the impact of routing instabilities by lowering routing convergence times; and controlling IGP paths by performing traffic engineering (i.e., adjusting the network configuration to control the flow of traffic inside the network). We now summarize the main efforts in each of these areas.

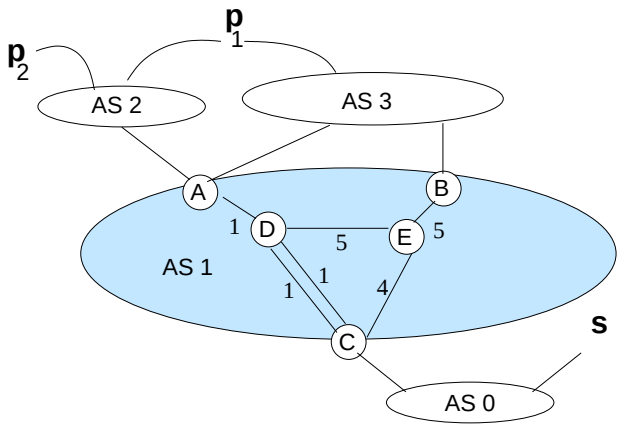


Figure 2.3: Example of an internetwork with four ASes to illustrate the impact of link failures on routing.

2.2.1 Monitoring

Understanding the dynamics of intradomain routing is essential for managing a network. Not surprisingly, there has been both commercial and research interest in developing monitors to track IGP behavior [IPS, Pkt, IPM, WLJ03, SG04]. These monitors are mainly used for analyzing the health of the network. In particular, IGP monitoring can pinpoint and characterize routing anomalies such as routing loops and flapping links [LAJ99, SIG+02, InCM+02, WLJ03].

IGP monitors listen, timestamp, and store IGP messages. Since link-state protocols use reliable flooding, even the deployment of only one monitor allows it to capture all messages with intradomain routing changes. The monitoring system can then use these messages to reconstruct the state of the routing table at any router in the network. If the network is partitioned in areas, then the monitor sees detailed topology changes in its area. It only receives summary messages from other areas. Yet, this information is sufficient for reconstructing the routing table of any router in the same area as the monitor. Our measurement of hot-potato routing uses the OSPF monitor described in [SG04]. This monitor is located in the same physical location of an operational router

in a large ISP network. It forms an OSPF adjacency with this router to log the receipt of OSPF messages.

2.2.2 Routing convergence

Equipment failures and planned maintenance inside an AS trigger IGP routing changes. Upon receiving a message reporting a change, routers need to recompute their best paths to all other routers and update their forwarding tables. The period of time between when the change happens and the last router updates its routing information is called *routing convergence*. During routing convergence packets may be caught in loops, which may cause them to be delayed, received out of order, and ultimately lost.

Driven by customer demands for predictable performance, the network operations community has dedicated considerable effort to reducing the convergence time after an IGP routing change to a few seconds [FV00, AJY00, ICBD04]. They achieve faster convergence through two different approaches: faster failure detection through routers with much smaller keep-alive timers and link technologies that directly notify the router of failures; and faster recovery by splitting traffic over multiple shortest paths.

For example, Figure 2.3 represents an internetwork with four ASes, and the intradomain topology of AS 1 with routers A, B, C, D, E and edge labels representing the IGP weight of the link. When one of the links between C and D fails, packets from s to both p_1 and p_2 are lost for less than a second [ICBD04] before router C detects the failure and starts forwarding all packets using the remaining link. If the other link from C to D fails, then all routers need to update their forwarding tables. Although during IGP convergence packets may be delayed or lost in forwarding loops, a study of the Sprint backbone network found no transient loops associated with IGP changes in this network, which they attribute to the heavy use of equal-cost multiple paths [SMD03]. This lack of instability associated with intradomain routing changes suggests that in contemporary networks there is little disruption associated with IGP routing changes.

2.2.3 Traffic engineering

Traffic engineering—adapting the flow of traffic to the prevailing network conditions—is a common task. One mechanism that network administrators use to control the flow of traffic inside their network is to *tweak IGP weights*. Consider again the example in Figure 2.3. Suppose that after the failure of one of the links between C and D, the remaining link becomes congested. A network administrator can alleviate the congestion by tweaking the IGP weights to move traffic away from congested links [FGL+00, FRT02, NSTD03]. For instance, the administrator might decide to move the traffic destined to p_1 from egress point A to B by changing the weight on the link CD to 10. This action may also divert some of the traffic to p_2 to use the link CE, which may cause congestion. To make the traffic destined to p_1 exit at router B and keep that to p_2 using the link CD, the administrator also needs to increase the weight on the link DE. Changing IGP weights to alleviate congestion triggers IGP convergence, and may also trigger BGP convergence (as explained in Section 2.3.2).

This approach works in this simple example, but in general IGP-weight tweaking is too coarse grained. Large ISP networks have hundreds of routers and changing an IGP weight affects all destination prefixes together, therefore finding the optimal setting for the IGP weights is not an easy task. Research on traffic engineering has shown how to tune the configuration of IGP link weights [FT02, LORS01, FT00, FRT02, ERP02, BRRT03] to the prevailing traffic. However, the resulting optimization problems are NP complete, forcing the use of local-search techniques. Finding a good setting of the configurable parameters is especially difficult when routing must be robust to equipment failures [NSTD03, FT03, SG05]. These optimizations involve repeatedly running Dijkstra's all-pairs shortest paths algorithm to evaluate candidate settings of the IGP weights, and are only used offline. Further, tweaking IGP weights while trying to minimize the impact on interdomain routes is even more challenging: IGP distance changes are likely to trigger egress changes due to hot-potato routing.

2.3 Interdomain routing

The Border Gateway Protocol (BGP) [RL95, RLH04] is the interdomain routing in the Internet today. It is a path-vector protocol that allows each AS to apply local policies in selecting and propagating routes for each destination prefix. Network administrators use BGP policies to express business relationships between ISPs and customers, or to determine preference among choices of connections (for instance, to determine backup paths or to balance load).

Two routers establish a BGP session to exchange BGP messages over an underlying TCP connection. BGP routers send new update messages only when something has changed. An *advertisement* notifies a neighbor of a new or a modified route, whereas a *withdrawal* revokes a route that is no longer available. An advertisement may be a "replacement" of an earlier route (i.e., an *implicit withdrawal*) or a new "announcement" for a prefix. Each advertisement includes various route attributes, including the list of ASes along the path.

A large backbone network typically has multiple BGP-speaking routers, and BGP sessions with multiple neighboring ASes. Such a network can also have multiple BGP sessions with each neighbor AS. As a result, a router may receive routes for a destination prefix from multiple neighbors. The router applies *import policies* to filter unwanted routes and to manipulate the attributes of the remaining routes. The router then invokes a *decision process* to select exactly one "best" route for each destination prefix among all the routes learned from its neighbors. Different routers in an AS apply the BGP decision process independently and might select different "best" routes for the given prefix, depending on their locations in the network. For example, in Figure 2.3 router D selects the route via egress point A to reach p_1, whereas router E uses the route via B. Ultimately, each router applies *export policies* to manipulate attributes and decide whether to advertise the best route to each neighbor.

In addition to having *external* BGP (eBGP) sessions with neighboring ASes,

0. Ignore if egress point unreachable
1. Highest local preference
2. Lowest AS path length
3. Lowest origin type
4. Lowest Multiple-Exit Descriminator (MED), with same next-hop AS
5. eBGP-learned over iBGP-learned
6. Lowest IGP distance to egress point ("Hot potato")
7. Lowest router-id of BGP speaker

Table 2.1: Main steps in the BGP decision process.

routers may use BGP to distribute routing information within an AS. An *internal* BGP session operates in the same fashion as an eBGP session, with the exception that routes learned from one iBGP neighbor are not advertised to another iBGP neighbor. The iBGP sessions typically do not apply policies that manipulate the BGP attributes of the routes. In the example in Figure 2.3, routers A and B learn routes to p_1 from eBGP neighbors and propagate this information to router C via iBGP. Instead of having a full mesh of iBGP sessions, for scalability a large AS may introduce hierarchy through the use of *route reflectors* or *confederations* [HM01].

Ultimately, every router must select a single best route for each prefix among the advertisements from the various eBGP and iBGP neighbors. Table 2.1 summarizes the steps in the BGP decision process. Several steps depend on BGP attributes (such as local preference, AS path length, origin type, and Multiple-Exit Descriminator) that are conveyed in route advertisements and can be manipulated by local policies. However, IGP controls the two steps listed in bold-face. First, the router must determine if the BGP "next hop" (the "egress point") is reachable. Then, if multiple routes proceed through the next five steps of the decision process, BGP uses the IGP distance to select a route with the nearest egress point.

The fundamental role of interdomain routing in determining the performance of end-to-end paths in the Internet has made it an important research topic. We now discuss some efforts in monitoring and characterizing BGP dynamics (in particular, BGP

convergence properties), and in controlling the traffic flow between domains.

2.3.1 Monitoring

As with intradomain routing, monitoring BGP is important for understanding the dynamics of AS-level connectivity and end-to-end paths. RouteViews [RV] and RIPE-NCC [RIP] provide public data of passive measurements of BGP routing changes. Each of them logs the advertisement and withdrawal messages received via eBGP sessions with routers in participating ASes. These datasets are used by network administrators for debugging and troubleshooting purposes, and by researchers to characterize BGP dynamics under normal network conditions [GR97, LMJ98, LMJ99, LAJ99] or under stress (such as worm attacks [WZP+02]), AS-level connectivity [GR97, SARK02], and to help locate routing instabilities [MRWK03, FMM+04, CSK03].

An early characterization of BGP behavior [LMJ98] found significant level of routing instability due to bugs in router's operating system implementations. Although router vendors have since fixed these problems and the number of BGP updates has reduced by an order of magnitude, the number of route announcements remained large [LMJ99]. The large volume of BGP updates stems, in part, from the exploration of multiple alternate routes when a router switches from one best path to another [LABJ01, RWXZ02]. The next section discusses the impact of path exploration on convergence delays.

2.3.2 Routing Convergence

Understanding the properties of BGP routing convergence is challenging because route selection depends on locally-configured policies. Theoretically, ASes may set conflicting policies, which may lead to routing divergence [GW99]. Fortunately, under some constraints to local policies, BGP is guaranteed to converge to a unique solution [GR01].

BGP may take several minutes to converge after a topology or policy change [LABJ01, MGVK02], which is much higher than the time it takes for IGP convergence. BGP's long convergence time is mainly due to the exploration of alternate routes before selecting a stable route. Take again the example in Figure 2.1. When the link between UnivNet and Medium ISP fails, Small ISP will receive withdrawal messages from both Big and Large ISPs. However, there may be a delay between the reception of the two messages. Suppose that Small first receives the withdrawal from Big ISP. Then it will shift to use the route via Large, just to receive another withdrawal later. It will then withdrawal the route to the user. This example shows that the details of path exploration depend on timing details at routers throughout the Internet. Previous studies have shown that interarrival times of around 30 seconds are quite common for external routing changes, since Cisco routers use a 30-second minimum-route advertisement timer for eBGP sessions [LABJ01].

2.3.3 Traffic Engineering

Interdomain traffic engineering controls *inbound* and *outbound* traffic (or where traffic enters and exists the network, respectively). Network administrators have more control over outbound traffic than inbound [QUP+03]. When congestion happens at inter-AS links, ISPs tweak BGP policies for doing outbound traffic engineering [QUP+03, FBR03, Uhl04]. If the link between A and AS 2 is congested in Figure 2.3, administrators can change the BGP import policies to move traffic destined to p_1 away from A. In this case, the administrator can assign lower local preference to the route to p_1 learned from egress point A, which would cause C to use the route announced by B. Unfortunately, BGP tweaking is complex. The large number of destination prefixes may give enough flexibility for network administrators to achieve the desired traffic split just by tweaking BGP policies. Yet, determining this solution is not a trivial task. First, the search for the solution is usually done in an ad-hoc fashion. The search space

of all possible policies is extremely large and there is no obvious pattern for exploring it. Second, the solution needs to be expressed as a BGP policy, which involves writing route maps (e.g., regular expressions on BGP AS paths). Finally, any BGP optimization needs to model the effects of the IGP topology as well [FWR04]. BGP policies cannot be considered in isolation from IGP because of hot-potato routing as discussed in the next section.

BGP policy tweaking controls the choice of egress points, but not the choice of egress point per ingress router. For instance, in Figure 2.3 if the network operator modifies the import policies to give a lower priority to the route to p_1 learned at egress A, then all routers in AS 1 stop using A to forward traffic to p_1. BGP policy tweaking does not allow controlling the egress point that each router selects. If we consider the goal of minimizing disruption to egress-point selection, then the BGP tweaking is particularly ill-suited—it causes egress changes by design.

2.4 Interaction between Intradomain and Interdomain Routing

The previous sections explained the behavior of each tier of the routing system separately. However, the two tiers need to interact to ensure end-to-end packet delivery. When packets originate in an AS and destined to another, they are routed using information of both BGP and IGPs. BGP selects the AS path and, at each AS, the egress point, and IGPs select the path from ingress to egress inside each AS. In the example in Figure 1.1, C uses IGP information to (i) determine that B is the closest egress point and (ii) compute the outgoing link along a shortest path to B. The failure of router B, a link failure inside the network, or a change in IGP weights could cause router C to select the route from A. To summarize, the IGPs affect BGP in terms of:

- **Hot-potato routing:** As discussed earlier, the IGP distances affect the BGP decision process (presented in Table 2.1). If multiple BGP routes are equally good, the router selects the route with the closest egress point according to the IGP distance.

- **Next-hop reachability:** The IGP determines whether the routers in the AS believe that the egress point associated with the BGP route is reachable.

- **iBGP message delivery:** The iBGP sessions used to propagate BGP routes inside the AS depend on the IGP for message delivery. Transient packet loss during IGP routing convergence could trigger iBGP session failures.

- **Multi-exit discriminator:** An AS can use the MED attribute in BGP to specify exactly where traffic should enter the domain, a practice called *cold-potato routing*. When used, MED metrics are often tied directly to the AS's IGP distances, making internal IGP instability visible to neighboring domains. However, most ISPs only listen to the MED attribute of routes learned from customers, hence cold-potato routing is not as prevalent as hot-potato routing.

In this thesis, we focus on the impact of using hot-potato routing to select egress points in an AS. We do not investigate MED-based cold-potato routing. There has been no prior work that directly investigates the interaction between the two tiers of the routing system, and few that address the problem of egress-point selection. In particular, there is no work that explicitly studies mechanisms for selecting egress points. We now summarize related work in each broad area of this thesis.

2.4.1 Measurement-Based Characterization

As discussed earlier in this chapter, previous studies have characterized IGP link-state advertisements [WLJ03, SIG+02, InCM+02, LAJ99] or BGP update messages [LAJ99, RWXZ02, ACBD04, LMJ98] in isolation. This thesis is the first work to present a joint analysis of IGP and BGP messages. This joint analysis allows a deeper understanding of the behavior of the routing system. The behavior of a network cannot be fully explained by studying IGP alone, because BGP is responsible for determining the ingress and the egress points. On the other hand, BGP alone cannot explain the cause

of a perceived behavior, because any of the ASes in the Internet can be responsible for a BGP change.

Agarwal et al. [ACBD04] evaluate how BGP routing changes affect the flow of traffic inside an ISP backbone, but do not differentiate between routing changes caused by internal and external events. We will contrast our work with the one presented in [ACBD04] in more detail in Chapter 4. In addition, great attention has been given to both IGP [FV00, AJY00, ICBD04] and BGP [LABJ01, MGVK02] convergence in isolation. However, no prior work analyzes the dynamics of internal network events that trigger *BGP* routing changes, which we study in Section 4.1.1.

2.4.2 Model of Sensitivity to Internal Changes

Traffic engineering tools evaluate the impact of different network configurations on the traffic matrix. For example, Netscope [FGL+00] is a tool from AT&T Labs that allows network administrators to experiment with different IGP configurations to determine the load distribution across links. Although Netscope incorporates models of intradomain routing and hot-potato routing, these models are not formally described and hence cannot be reproduced. The algorithm presented in [FRT02] searches for the set of OSPF weights that leads to an optimal link load distribution. Subsequent work [FT02] considers weight settings that are more robust to link failures and changes in traffic demands. More recent work [FWR04] models the BGP routing decision in detail and allows the study of changes to BGP configuration on the egress-point selection. However, none of these tools focuses on the network sensitivity to internal routing changes specifically. They evaluate route selection and traffic distribution considering different network settings, but not the impact of the change on the network. Therefore, these tools cannot predict the sensitivity of the network to changes in the IGP topology. Chapter 5 presents a model of hot-potato routing and metrics to compute the network sensitivity to intradomain routing changes.

2.4.3 Egress-Point Selection

The study of active measurements presented in [SMA03] infers that many paths that use hot-potato routing are inflated if compared to "optimal" egress selection (i.e., the egress that leads to the shortest end-to-end path). This empirical result is in agreement with a game-theoretical analysis of hot-potato routing [JT03]. This analysis shows, however, that if we associate a cost per number of flows traversing each link, then the cost of hot-potato routing is bounded by three times the cost of shortest path routing.

Motivated by these findings, further research has considered how a pair of neighboring ASes could coordinate to select egress points in a mutually advantageous manner [MWA04, WJR02]. Whereas these papers focus on the negotiation process, and on the important question of what information the ASes should exchange, in Chapter 6 we propose a tunable mechanism for selecting the egress points and a way for each AS to determine its preferred egress points based on network-wide objectives.

In recent years, an increasing number of stub ASes, such as large enterprise and campus networks, connect to multiple upstream providers for improved reliability and flexibility. In response, several studies have considered how these networks should balance load over the multiple access links [AMS+03, GQX+04]. However, our problem is different because we focus on networks where each destination prefix has a (possibly different) set of egress points, and the choice of egress point affects the load on links *inside* the AS.

Chapter 3

Identifying Hot Potatoes

This chapter presents our methodology for identifying hot-potato routing changes from data routinely collected in large ISPs for management purposes. We define hot-potato changes and discuss the challenges associated with measuring it. After presenting our measurement infrastructure and our algorithm for identifying hot-potato changes, we describe our methodology for measuring the impact of BGP routing changes on traffic. The next chapter characterizes the impact of these hot-potato routing in an operational network.

On the surface, we should be able to study hot-potato routing changes in an analytical or simulation model based on the protocol specifications. However, the interaction between IGPs and BGP depends on details not captured in the IETF standards documents. Vendor implementation decisions have a significant impact on the timing of messages within each protocol. The design of the network, such as the number and location of BGP sessions, may also play an important role. In addition, the behavior of the routing protocols depends on the kinds of low-level events—failures, traffic engineering, and planned maintenance—that trigger IGP path changes, and the properties of these events are not well understood. In light of these issues, this thesis first takes an empirical approach of a joint analysis of IGP and BGP measurements collected from a large ISP network.

3.1 Hot-Potato Routing Changes

In this section, we present a precise definition of a "hot-potato routing change" and explain why identifying these routing changes in an operational network is surprisingly difficult.

3.1.1 Definition

The BGP decision process [RLH04] on a router selects a single best route for each destination prefix by comparing attribute values as discussed in Section 2.3. Two of the steps depend on the IGP information. First, BGP excludes any route for which the BGP next-hop address is not reachable. For example, in Figure 1.1, the router C does not consider the BGP route from A if C's forwarding table does not have an entry that matches A's IP address (i.e., if C cannot reach A). Then, after the next five steps in the decision process, the router compares IGP distances associated with the BGP next-hop addresses and selects the route with the smallest distance. If multiple routes have the same IGP distance, the router applies additional steps to break the tie. Hot-potato routing is when the BGP decision process is decided by comparing the IGP distances to the candidate egress points (i.e., step 6 in Table 2.1). When a change in an IGP distance leads a router to select a different egress point, we refer to this as a *hot-potato routing change*.

To guide our characterization of hot-potato routing, we propose a simple model that captures the path selection process at a single router (which we denote as a *vantage point*):

- **Distance vector (per vantage point)**: The vantage point has a distance vector that represents the cost of the shortest IGP path to every router in the AS. The distance vector, which changes in response to link failures and modifications to the link weights, is a concise representation of the aspects of the IGP that can affect BGP routing decisions at this vantage point.

- **Egress set (per prefix):** The network has a set of routers that have eBGP-learned routes that are the "best" through step 5 in the BGP decision process. These routes can be propagated to other routers in the AS via iBGP.

For each prefix, the vantage point selects the egress point (from the egress set) with the smallest distance (from the distance vector). A hot-potato routing change occurs when a vantage point selects a different egress point because of a change in the distance vector (i.e., that makes the new egress point closer than the old one). For example, initially router C in Figure 1.1 has an egress set of $\{A, B\}$, distance vector $(A \rightarrow 9, B \rightarrow 10)$, and a best egress point of A; then, when the IGP distance to A changes to 11, C selects egress point B. Our goal is to *determine what fraction of the BGP routing changes are hot-potato routing changes in an operational network*.

3.1.2 Challenges of Characterizing Hot-Potato Routing

On the surface, measuring the hot-potato routing changes seems relatively simple: collect BGP and IGP measurements from an operational router and determine which BGP messages were triggered by IGP routing changes. However, several factors conspire to make the problem extremely challenging:

Incomplete measurement data

A large operational network usually has hundreds of routers, fully instrumenting all of them is not possible; instead, we must work with data from a limited number of vantage points. In addition, commercial routers offer limited opportunities for collecting detailed routing data—we can only collect measurements of the routing protocol messages that the routers exchange among themselves.

IGP measurements are difficult to collect since they often require a physical connection to a router located in a secure facility. Fortunately, in link-state protocols like OSPF and IS-IS, the routers *flood* the link-state advertisements (LSAs) throughout

the network, allowing us to use data collected at one location to reconstruct the distance changes seen at other routers in the network. However, this reconstruction is not perfect because of delays in propagating the LSA from the point of a link failure or weight change to other routers in the network.

Collecting BGP data from multiple routers is easier because BGP sessions run over TCP connections that do not require a physical adjacency. However, BGP messages from the operational router must traverse the network to reach the collection machine, which introduces latency. Further, these delays may increase precisely when the IGP routes are changing. In addition, since BGP is a path-vector protocol, a router sends *only its best route* to its BGP neighbors, making it difficult to know the complete set of routing choices a router has available at any given time.

Complex routing protocol dynamics

IGP routing changes stem from topology changes (i.e., equipment going up or down) and configuration changes (i.e., adjustments to the link weights). Monitoring the IGP messages shows only the *effects* of these events. In practice, multiple LSAs may occur close together in time (e.g., the failure of a single router or an optical amplifier could cause several IP links to fail). If one LSA follows close on the heels of another, the routing system does not have time to converge after the first LSA before the next one occurs.

Similarly, a prefix may experience multiple BGP routing changes in a short period of time (e.g., a neighboring AS may send multiple updates as part of exploring alternate paths). A hot-potato routing change might also trigger multiple iBGP routing changes as the network converges. In addition, the global routing system generates a constant churn of BGP updates due to failures, policy changes, and (perhaps) persistent oscillations. Continuously receiving several updates a second is not uncommon. This large number of BGP updates makes it difficult to identify which BGP routing changes

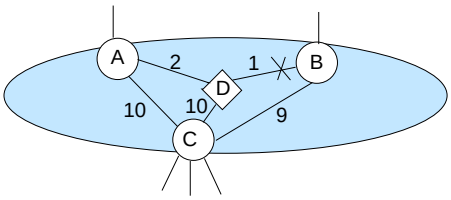


Figure 3.1: Router C changes best route without IGP distance change.

are caused by hot-potato routing inside the AS. The Multiple-Exit Discriminator (MED) attribute introduces additional complexity because the BGP decision process compares MED values only across routes learned from the same next-hop AS, resulting in scenarios where a router's local ranking of two routes may depend on the presence or absence of a third route [GW02a].

Hierarchy of iBGP sessions inside the AS

Large networks often employ *route reflectors* to reduce the overhead of distributing BGP information throughout the AS [HM01]. However, route reflectors make the dynamics of network-wide routing changes more complicated. In the example in Figure 3.1, router D is a route reflector with clients A, B, and C. Both C and D have shorter IGP paths to B than A. When the BD link fails, router D shifts its routes from egress B to egress A. However, since C is a client of D, it too would change its routes to use egress A even though its own distance vector has not changed! Determining which BGP routes from C are caused by IGP changes requires knowing the route-reflector configuration of the network and which BGP routing changes from D were caused by the IGP. Some *under-counting* of hot-potato routing changes is inevitable, though focusing the analysis on the "top-level" route reflectors in the network helps limit these effects.

Vendor implementation details

Although the routing protocols have been standardized by the IETF, many low-level details depend on implementation decisions and configuration choices. For example, the final tie-breaks in the BGP decision process vary from vendor to vendor. The vendor implementations have numerous timers that control when the router recomputes the IGP paths, reruns the BGP decision process, and sends update messages to BGP neighbors. The router operating system may have complex techniques for scheduling and preempting tasks when multiple events occur close together in time. These router-level details can have a first-order impact on the network-wide dynamics of hot-potato routing.

Together, these issues suggest that computing an exact measure of hot-potato routing changes is extremely difficult. Fortunately, we can use reasonable heuristics to find approximate numbers.

3.2 Measuring Impact on the Control Plane

In this section, we present our methodology for measuring hot-potato changes experienced by operational routers. Figure 3.2 presents the steps to correlate BGP updates from a vantage point with IGP messages. Each dotted box represents steps described in a particular subsection. Section 3.2.1 presents the measurement infrastructure used to collect BGP updates and IGP messages. We describe how to compute the distance vector from the IGP messages in Section 3.2.2. Section 3.2.3 explains the classification of BGP routing changes in terms of the possible causes. This sets the stage for the discussion in Section 3.2.4 about how to associate BGP routing changes with related distance changes that occur close in time.

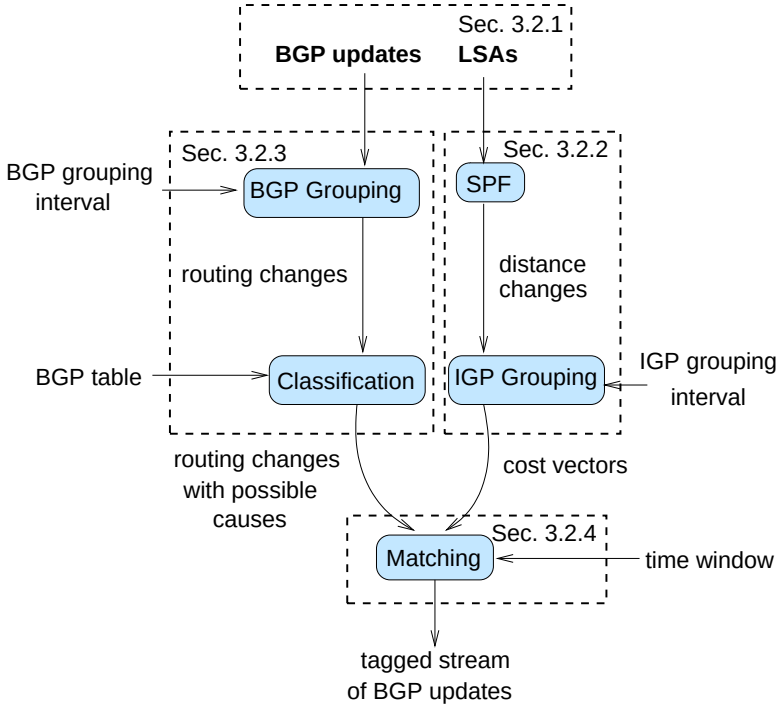


Figure 3.2: Steps for identifying hot-potato routing changes. Dotted boxes are labeled with the number of the subsection that describes it.

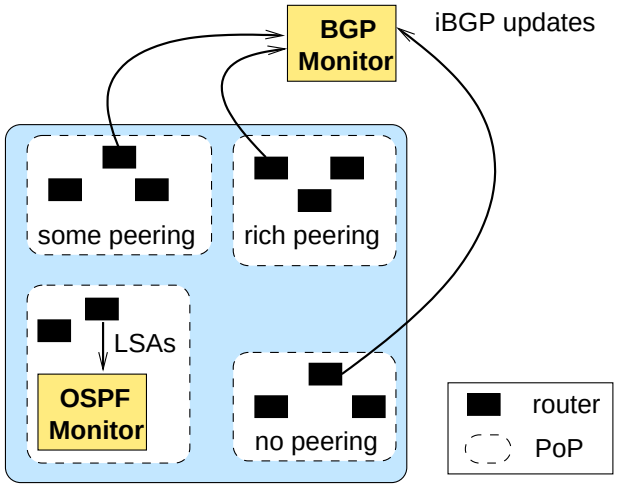


Figure 3.3: Measurement infrastructure in AT&T backbone.

3.2.1 Measurement Infrastructure

In this thesis, we study AT&T's tier-1 backbone network (AS 7018), which uses OSPF as intradomain routing protocol. We have deployed an OSPF and a BGP monitor in this network. Figure 3.3 depicts our measurement infrastructure. The OSPF monitor [SG04] is located at a Point of Presence (PoP) and has a direct physical connection to a router in the network. AT&T uses OSPF with areas. We connect our monitor to a router in area 0 to have complete view of the distances to reach each router. The monitor timestamps and archives all LSAs. The BGP monitor has iBGP sessions (running over TCP) to several top-level route reflectors. Using an *iBGP* session allows the monitor to see changes in the "egress point" of BGP routes. The BGP monitor also dumps a snapshot of its routes four times a day to provide an initial view of the best route for each prefix for each vantage point. The OSPF and BGP monitors run on two distinct servers and timestamp the routing messages with their own local clocks; to minimize timing discrepancies, both monitors are NTP synchronized.

Our analysis focuses on 176 days of data collected from January 2003 to July 2003. Because details of network topology, peering connectivity, and the absolute num-

ber of routing messages are proprietary, we omit router locations and normalize most of our numerical results. We study data collected from three vantage points—all Cisco routers that are top-level route reflectors in different PoPs. To explore the effects of router location and connectivity, we select three vantage points in PoPs with different properties:

- *Rich peering* is a router in a PoP that connects to a large number of peers, including most major ISPs.
- *Some peering* is a router in a PoP that connects to some but not all major peers.
- *No peering* is a router in a PoP with no peering connections. Most traffic is directed to egress points in two nearby PoPs.

The three PoPs are located in the eastern part of the United States, relatively close to the location of the two route monitors.

Resets of the monitoring session would affect the accuracy of our results, especially if IGP routing changes are correlated with iBGP session resets. Each of the BGP monitoring sessions experienced at most five resets per month, perhaps due to temporary disruption of the monitor's connection to the rest of the network. These results suggest that IGP events were not a significant contributor to iBGP session resets in the network. In fact, the default keep-alive and hold timers for BGP sessions (60 and 180 seconds, respectively) make it unlikely that transient disruptions during IGP convergence would affect iBGP reachability. Before conducting our analysis, we eliminate all destination prefixes where the BGP routing decisions depend on MEDs. To be conservative, we exclude any prefix that had *any* BGP update with a non-zero MED attribute during the period of the data collection, which represent approximately 13% of the total number of prefixes.

3.2.2 Computing Distance Vector Changes

OSPF is a link-state routing protocol where each unidirectional link is assigned an administrative weight that is flooded throughout the network in a reliable fashion [Moy98]. The OSPF monitor has an algorithm to process the LSAs as they arrive to continuously track the OSPF topology and compute the distance vector changes from each vantage point [SG04]. First, the algorithm disregards any LSAs that do not reflect a change in the OSPF topology; this process excludes OSPF's periodic refresh LSAs as well as any duplicate LSAs sent in the reliable flooding process. For the remaining LSAs, it emulates the OSPF shortest-path computation [Moy98] to determine the distance from each vantage point to every other router at the boundary of the network (i.e., any router that could serve as an egress point for one or more prefixes).

Some OSPF topology changes do not trigger distance changes. For example, some links with high OSPF weights do not appear on any shortest path (e.g., links under maintenance or provisioning); an increase in the weight or the failure of the link would not affect any of the shortest paths. Also, some links always appear as part of multiple shortest paths along with other links (e.g., parallel links between two routers). Other LSAs may change the distances for one vantage point but not another. Whenever an LSA changes one or more distances for a given vantage point, the algorithm produces a new distance vector for that vantage point. If the vantage point cannot reach another router (e.g., due to a failure or network partition), we represent the distance as ∞. Our study focuses on the common case of distance changes from one finite value to another.

In practice, multiple LSAs may occur close together in time. Even if these LSAs stem from different events (e.g., two independent failures), the delays in propagating the LSAs and in converging to new routes make it impossible to analyze these LSAs separately. Instead, we group distance changes that occur within a small time window into a single distance vector change. We select the interval duration based on analysis of our OSPF measurements, shown by the "distance changes" CDF in Figure 3.4. To gen-

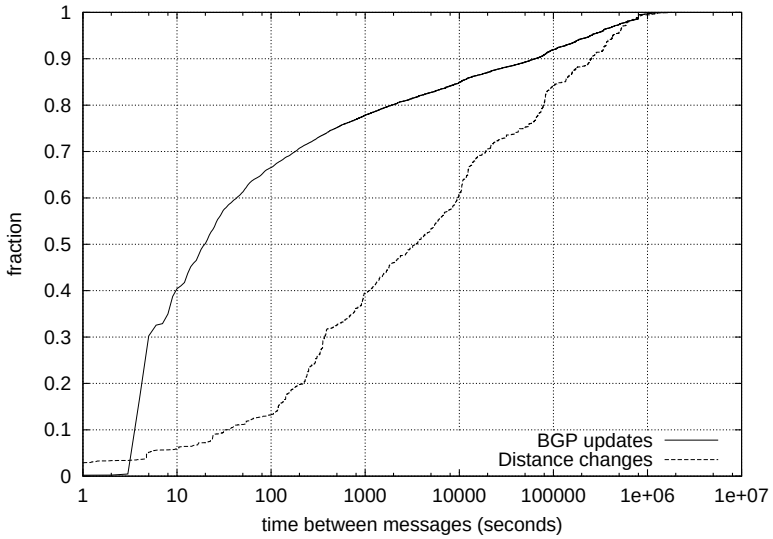


Figure 3.4: CDF of message interarrivals for each protocol.

erate this CDF, we consider the interarrival times of the distance changes between each vantage point and all possible egress routers and plot the resulting cumulative distribution. About 5% of the distance changes occur within ten seconds of each other. These may correspond to LSAs caused by a single physical event, such as rebooting a router. Otherwise, the curve increases gradually over the range of values. Half of the distance changes have an interarrival time of more than 3400 seconds, and 10% are more than 252,000 seconds (almost a month). In the next chapter, we apply a time interval of 10 seconds for grouping distance changes; additional experiments showed that the results were not sensitive to small changes in the size of the interval.

3.2.3 Classifying BGP Routing Changes

The global BGP routing system generates a continuous stream of update messages, as shown by the example in Figure 3.5. This graph plots the number of BGP updates (left y-axis) and distance changes (right y-axis) seen by the "rich peering" router over one hour, with one-minute bins. In this example, the router sometimes makes sev-

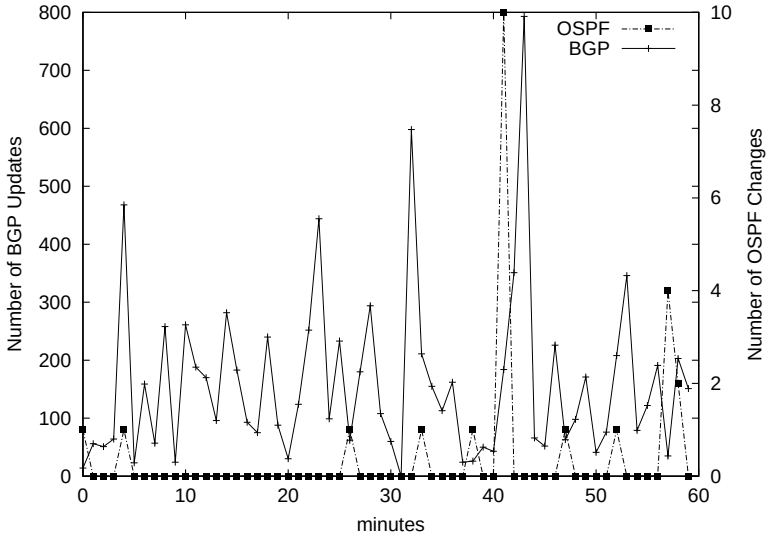


Figure 3.5: Time series of BGP updates and distance changes.

eral hundred BGP routing changes in a minute. In contrast, very few intervals have more than a handful of path cost changes, and these changes do not necessarily cause the router to switch from one egress point to another for any prefix. As discussed in Section 2.3, the large volume of BGP updates is caused by the exploration of alternate routes during convergence. These short-lived BGP routes do not correspond to stable path changes but rather the *transition* from one stable route to another. In contrast, in this thesis, we are interested in how IGP distance changes cause a router inside the AS to switch from one stable route to another with a different egress point.

To focus on changes from one stable route to another, we group BGP updates at the same router for the same prefix that occur close together in time, based on the "BGP updates" curve in Figure 3.4. To generate the curve, we consider the interarrival times of the BGP updates from each vantage point for each prefix and plot the resulting cumulative distribution. More than 30% of the BGP updates have an interarrival time of five seconds or less. This stems from the 5-second minimum-route advertisement timer used by Cisco routers to pace the update messages on iBGP sessions. Previous studies

have shown that interarrival times of around 30 seconds are quite common for external routing changes, since Cisco routers use a 30-second minimum-route advertisement timer for eBGP sessions [LABJ01]. In Figure 3.4 about two-thirds of the BGP updates have a spacing of less than 70 seconds. In the next chapter, we apply a time interval of 70 seconds for grouping BGP messages to combine many of the transient updates occurring during path exploration. Additional experiments showed that the results were not sensitive to small changes in the size of the grouping interval.

Many BGP routing changes have no relationship to the distance vector changes in the interior of the network. Drawing on the BGP decision process, our algorithm classifies BGP routing changes in terms of their possible causes. Starting with an initial BGP routing table, we consider a stream of changes in the best route for each prefix. Determining why a router changes from one route to another is difficult in practice, since multiple events may trigger the same BGP update message. The root cause is not necessarily an IGP event. New advertisements from neighboring ASes and changes in local routing policies can cause changes in the best path as well. For example, suppose that a router switches its best route for a destination prefix from r to s. One possibility is that an eBGP neighbor started advertising a more attractive route s (say, with a shorter AS path); another possibility is that an increase in the IGP path distance to r's egress point caused the router to prefer a closer egress point with route s.

To aid the analysis, we propose a classification of BGP update messages that identifies the types of OSPF events that *could* explain a change we see in the BGP-level routing decision. Figure 3.6 illustrates how we classify a BGP routing change from route r to route s for a prefix at a particular vantage point. Hot-potato routing changes cause a router to switch from one BGP route to another. As such, changing from or to a null route does not represent a hot-potato routing change. However, hot-potato routing changes can cause s to *replace* r. In this case, further analysis helps narrow down the possible causes. If r and s have the same egress point, a change in the distance vector

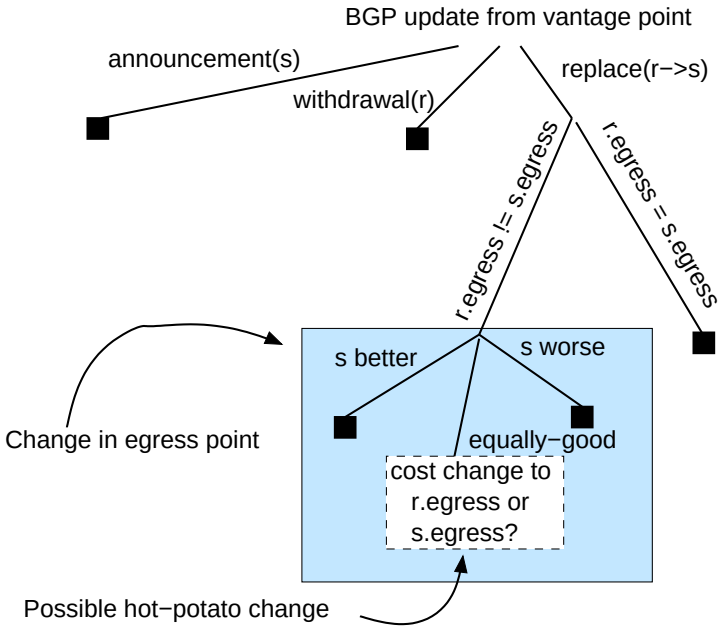


Figure 3.6: Classifying BGP routing changes at a vantage point. Black boxes are terminals indicating that the BGP update is *not* a hot-potato routing change.

cannot be responsible.

Having different egress points $r.egress$ and $s.egress$ does not necessarily imply that hot-potato routing is responsible. The new route s might be "better" than the old one at some earlier stage in the decision process; for example, s might have a shorter AS path or a larger local-preference. Alternatively, the route r might have been withdrawn; because our monitor sees only the best route at each vantage point, we can only infer that r was withdrawn if s is "worse" than r. Hence, if r and s are not "equally good" through steps 0–5 of the BGP decision process, we can dismiss hot-potato routing as a possible cause. If the routes are equally good, hot-potato routing *might* be responsible if the relative "closeness" of the two egress points has changed—making the egress point s closer than egress point r.

3.2.4 Matching Distance Changes with BGP

To further refine our inference that an IGP routing change caused the vantage point to select s, we inspect the stream of distance vectors for this vantage point to see if $s.egress$ became closer than $r.egress$ within some small time interval. We verified the correctness of this algorithm in a controlled environment using the testbed presented in [TSGR04]. In this scenario, all BGP routes are stable and the only changes are related to distance changes; our algorithm correctly identified the OSPF LSA that caused each BGP update. However, BGP routes are *not* stable in the operational network. Hence, our algorithm might mistakenly match a BGP routing change with an *unrelated* distance vector change. The BGP routing change might have been triggered by an external event, such as a policy change or a failure in another AS, that caused r to be withdrawn or replaced by a less attractive route. Yet, a seemingly-related distance vector change could occur nearby in time that is consistent with the vantage point's decision to switch to route s. In this situation, our algorithm would mistakenly associate the replacement of r by s with the distance change. (In practice, the IGP event might have caused a similar BGP routing change anyway if the external event had not happened first.)

Although these kinds of mismatches are difficult to avoid completely, three aspects of our algorithm reduce the likelihood of false matches: (i) preprocessing the distance vector changes and BGP update messages as discussed in Section 3.2.2 and 3.2.3, (ii) the fine-grained classification in Figure 3.6 which eliminates many of the external BGP routing changes, and (iii) the careful selection of the time window for correlating the two datasets. To find the appropriate time window, we first consider distance vector changes that occur within ten minutes before or after the BGP routing change. Although our algorithm did find occasional matches over the entire 20-minute interval, the vast majority of hot-potato BGP routing changes occurred within *three minutes* of the distance vector change, for reasons we explore in more detail in the next chapter. In experiments where we did *not* preprocess the OSPF and BGP data, we tended to see a

larger number of (presumably false) matches in the large time intervals, suggesting that our preprocessing is helpful for reducing the likelihood of false matches.

Our algorithm finds some matches where the BGP routing change appears to happen 1–2 seconds *before* the distance vector change. Although this seems counter-intuitive, this can occur in practice for two reasons. First, the OSPF LSA may take longer to reach our OSPF monitor than for the related BGP update to reach the BGP monitor. The reliable flooding of OSPF LSAs is typically implemented in software on the router, which may subject these messages to higher delays. In contrast, BGP update messages are sent via a TCP connection between two routers; the IP packets carrying these messages traverse the hardware forwarding path through the routers. Second, the BGP monitor has a coarser timestamp resolution than the OSPF monitor. To account for these two issues, we allow a small *negative* time difference between the distance vector change and the BGP change. Therefore, in practice we have found that a time window of $(-2, 180)$ is a reasonable way to avoid false matches while still capturing the bulk of the real hot-potato routing changes. We use this window for the analysis in the rest of the thesis.

3.3 Measuring Impact on the Data Plane

This section presents our methodology for measuring the impact of egress changes on traffic. The *traffic matrix* TM represents the volume of traffic from an ingress point to an egress point. The traffic matrix is the composition of the *traffic demands* and the *egress point selection*. We represent the traffic demands during a time interval t as a matrix $\mathcal{M}$, where each element $\mathcal{M}(i, p, t)$ represents the volume of traffic entering at ingress router i headed toward a destination prefix p. We represent the BGP routing choice as a mapping ε from a prefix to an egress point, where $\varepsilon(i, p, t)$ represents the egress router chosen by ingress router i for sending traffic toward destination p. At time t each element of the traffic matrix TM is defined as:

$$TM(i, e, t) = \sum_{p \in P : \varepsilon(i,p,t) = e} \mathcal{M}(i, p, t). \tag{3.1}$$

where P is the set of all destination prefixes.

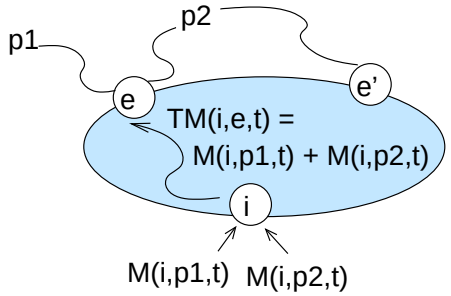


Figure 3.7: Example of traffic matrix.

Figure 3.7 presents a simple network with one ingress router i, two egress routers e and e', and two external destination prefixes p_1 and p_2. Given traffic demands $\mathcal{M}(i, p_1, t)$ and $\mathcal{M}(i, p_2, t)$ and a prefix-to-egress mapping $\varepsilon(i, p_1, t) = \varepsilon(i, p_2, t) = e$, the traffic matrix for this network is $TM(i, e, t) = \mathcal{M}(i, p_1, t) + \mathcal{M}(i, p_2, t)$ and $TM(i, e', t) = 0$.

3.3.1 Measurement Infrastructure

Studying the impact of egress changes in the traffic matrix elements over time requires collecting fine-grained measurements of traffic and routing. We collect data from eight aggregation routers that receive traffic from customers destined to peers and other customers. The eight routers are located in major Points of Presence (PoPs) that are spread throughout the United States. The “no peering”, “some peering”, and “rich peering” routers are among the eight routers studied.

Figure 3.8 provides a high-level view of our measurement infrastructure and how we use the data to compute part of the ingress router to egress PoP traffic matrix

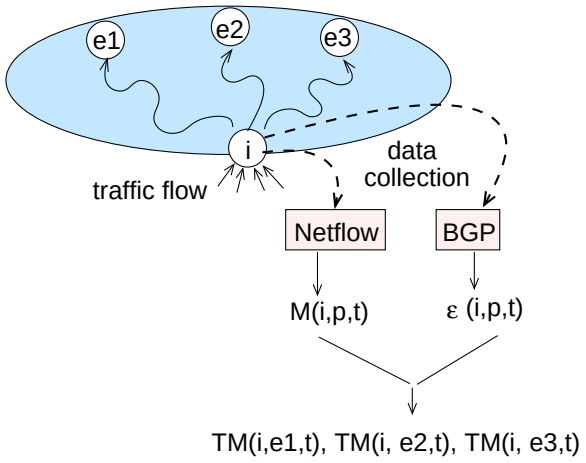


Figure 3.8: Measurement of routing and traffic data from an ingress point i.

(the ingress router i represents one of our eight vantage points). We choose to measure the *PoP-level* traffic matrix because routers in the same PoP essentially act as one larger node. We compute eight *rows* of the traffic matrix, considering all traffic from these eight ingress aggregation routers to all of the egress PoPs. This section describes how we compute the prefix-to-egress mapping $\varepsilon(i, p, t)$ from the BGP data and the traffic demands $\mathcal{M}(i, p, t)$ from the Netflow [Cisf] data. Once we have computed ε and $\mathcal{M}$, we use Equation 3.1 to compute the elements of the traffic matrix $TM(i, e, t)$. The BGP monitor and the Netflow collection servers are NTP-synchronized, allowing us to use the timestamps to join the two datasets.

Prefix-to-Egress Mapping

We use the BGP monitor introduced in Section 3.2.1 to collect iBGP update messages directly from each vantage point. Again, we group the BGP updates for the same destination prefix that have an interarrival time of 70 seconds or less to focus on stable routing changes. Based on an initial BGP table dump and a sequence of BGP updates, we generate the prefix-to-egress mapping $\varepsilon(i, p, t)$ for any given time. The

egress point corresponds to a *PoP* rather than a specific router. We associate each egress router with a PoP based on the router name and configuration data.

Traffic Demands

Every vantage point has Cisco's Sampled Netflow feature [Cisf] enabled on all links that connect to access routers and exports flow records to a collection server at the same location. The collection server samples the flow records using the technique presented in [DLT03] to reduce processing overhead, and computes 10-minute aggregated traffic volumes for each destination prefix. We use these aggregated reports to extract $\mathcal{M}(i, p, t)$ for each vantage point i and destination prefix p at every 10-minute interval. Consequently, a reference to a time t indicates the end of a 10-minute interval.

Because of sampling, the volumes $\mathcal{M}(i, p, t)$ are random quantities that depend on the sampling outcomes. Through renormalization applied to the bytes reported in sampled flow records, the quantities $\mathcal{M}(i, p, t)$ are actually unbiased estimators of the volumes of the original traffic from which they were sampled, i.e., their average over all possible sampling outcomes is the original volume. The standard error associated with an aggregate of size $\mathcal{M}$ is bounded above by $\sqrt{k/V}$ for some constant k that depends on the sampling parameters [DLT03]. For the parameters employed in this thesis, $k < 21$MB. Note that the standard error bound decreases as the size of the aggregate increases. The largest changes in traffic rates are the most reliably estimated. As an example, for a 10-minute aggregate of traffic at a rate of 10 MB per second, the standard error due to sampling is no more than 6%. The higher accuracy for larger traffic rates aligns well with the focus of Section 4.2 on the largest traffic matrix variations.

Even though the traffic data is divided into 10-minute intervals, our 70-second grouping of BGP updates is important for cases when path exploration crosses the boundary between two ten-minute intervals. This ensures that we focus our analysis on stable changes of ε. If the mapping $\varepsilon(i, p, t)$ changes more than once in a 10-minute

interval, then we cannot distinguish the volume of traffic affected by each of them individually. Therefore, we exclude those cases from our analysis by ignoring intervals with prefixes that have more than one stable routing changes in that bin; this excludes only 0.05% of the (i, e, t) tuples from our study. We also exclude all traffic for the small number of flows that had no matching destination prefix in the BGP routing tables or update messages; we verified that these flows corresponded to an infinitesimal fraction of the traffic.

3.4 Summary

In this chapter, we presented a methodology for joint analysis of OSPF and BGP measurement data. We describe an algorithm that replays a log of routing messages received from an operational router to identify which of a stream of BGP updates are caused by hot-potato routing. Our algorithm is divided into three main steps: (i) process OSPF messages to obtain a distance vector for each vantage point; (ii) classify BGP updates received from the vantage point according to possible cause; and (iii) match both streams of routing messages in time. This algorithm allows measuring the impact of internal routing changes on BGP routing. Then, we extend our measurement methodology to study the impact of routing changes on traffic. We combine a continuous view of both Netflow and BGP data to study variations on the PoP-to-PoP traffic matrix. The next chapter presents the results of applying this methodology to a Tier-1 ISP network.

Chapter 4

Impact of Hot-Potato Changes

This chapter presents a case study of hot-potato routing changes in an operational network. Our goal is to identify and understand the main properties of hot-potato routing changes, rather than to highlight specific numerical values that might vary from one network to another. First, we analyze the impact of using hot-potato routing on BGP egress point selection. We find that, although most hot-potato routing changes occur within 60 seconds, extra delays of 1–2 minutes sometimes arise due to the iBGP hierarchy and the transfer of update messages. Further, the frequency of hot-potato routing changes varies significantly across time and router location. Interestingly, the hot-potato BGP updates have a much more even spread across the destination prefixes than the remaining update messages. Then, we study the impact on traffic. Hot-potato changes are responsible for the largest variations in the traffic matrix.

4.1 Control Plane

This section characterizes the dynamics of hot-potato changes on the routing control plane. We use the methodology presented in the previous chapter to match BGP and OSPF routing changes. First, we study the time it takes for the routing system to converge after an intradomain routing change. This time can be divided into (i) the time

for the BGP process in a router to learn about the distance change, and (ii) the time BGP takes to update all the destination prefixes that change egresses because of the distance change. Then, we characterize the frequency of hot-potato changes across different router locations, days, and destination prefixes. Finally, we measure the number of prefixes that shift egress points because of an internal routing change.

4.1.1 BGP Reaction Time to Distance Changes

Figure 4.1 presents the cumulative distribution of the delay between a distance vector change and a correlated BGP routing change for the "no peering" router from January 2003 to July 2003. For comparison purposes, this graph also presents the lab results from [TSGR04], which was generated from a controlled experiment with a router in a lab. For this experiment all BGP routing changes were caused by an OSPF routing change. The lab curve is almost perfectly linear in the range of 5 to 65 seconds due to the influence of two timers. First, the router imposes a 5-second delay after receiving an LSA before performing the shortest-path computation to avoid multiple computations when several LSAs arrive in a short period of time [Cisd]. A second LSA that arrives during this interval does not incur the entire five-second delay, as evidenced by the small fraction of LSAs that experienced less than five seconds of delay. Second, the router has a 60-second scan timer that runs periodically to sequence through the BGP routing table and run the BGP decision process for each prefix [Cisg]. The BGP change does not occur until the scan process runs and revisits the BGP routing decision for this prefix. As such, the delay in the BGP routing change is uniform in $[5, 65]$, as evidenced by the straight line in the graph. A router also imposes a 10-second interval between two consecutive shortest-path calculations, which explains delays in the $[65, 70]$ range.

The graph shows a significant gap between the results for the lab experiments and the curve for *all* hot-potato changes sent by the "no peering" router. Upon receiving a new LSA, the router must (i) rerun the IGP shortest-path computation, (ii) apply the

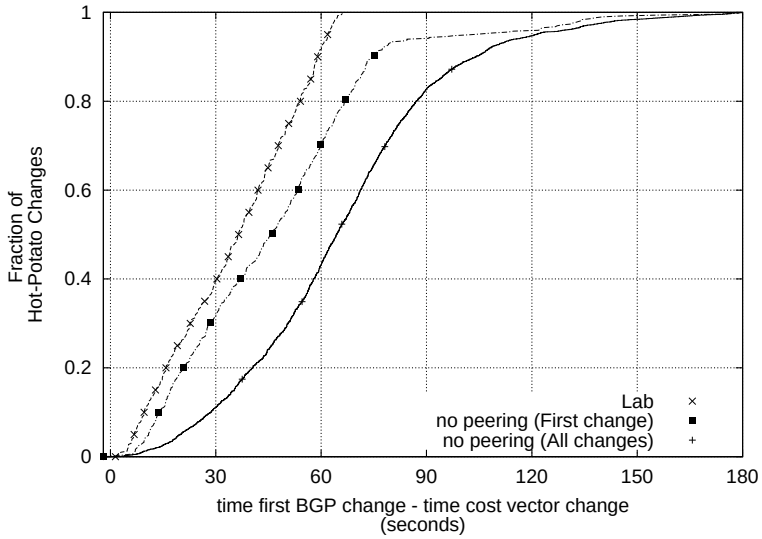


Figure 4.1: CDF of time lag between the distance vector change and related BGP routing changes, using a 10-second window to group OSPF LSAs, a 70-second window to group the BGP update messages, and a $(-2, 180)$ window to correlate the distance vector changes with BGP routing changes.

BGP decision process to select the best route for each prefix, and (iii) send update messages to BGP neighbors for the routes that have changed. The first two steps represent the time required to react to a distance vector change, and the third step depends on the number of BGP routing changes. The lab experiments evaluated only the first two steps. To have a fair comparison, we also measure and report the delay between the distance vector change and the *first* prefix experiencing a hot-potato routing change.

The graph shows that most hot-potato routing changes occur within 80 seconds of the distance vector change, which is closer to the 70 seconds upper limit of the lab results. The extra 10 seconds are explained by the rate of LSA arrivals and the number of routes in an operational router. When the rate of LSAs is higher, the likelihood of incurring the 10-second delay between consecutive shortest-path calculations increases. The scan process may require several seconds in an operational router because of the

large number of BGP routes. The 60-second timer restarts after the *completion* of the previous scan; hence, the BGP reaction time also includes the time for the running the scan process. These two factors contribute to longer reaction times in the operational router. We discuss the reaction times longer than 80 seconds in the next subsection.

4.1.2 Transfer Delay for Multiple Prefixes

The difference between the curve for *all* hot-potato changes and the one for the *first* change corresponds to the delay to transfer BGP updates for multiple prefixes. When a distance vector change affects a large number of prefixes, the transmission of the BGP update messages to iBGP and eBGP neighbors introduces additional delay. We illustrate the effect of this additional delay in Figure 4.2. This graph presents the cumulative distribution of the number of hot-potato changes triggered by two distinct distance vector changes (the distance changes that affected the *largest* number of prefixes for the "no peering" and "some peering" routers during June 2003). Although the BGP update for the *first* prefix appears within 80 seconds of the distance vector change, some updates appear much later. For example, in the "no peering" curve, a single distance vector change affected the BGP routes for more than $80,000$ prefixes. Although the BGP change for the first prefix occurs 66 seconds after the distance vector change, the routing change for the last prefix occurred 83 seconds later, 149 seconds after the OSPF change.

The shape of this curve is mainly determined by the volume of data and the TCP transmission rate between the vantage point and the BGP monitor. In our experiments, the BGP monitor is within a few hundred miles of the "no peering" router and the update packets travel just a few hops through the network. Longer delays might be possible over iBGP sessions between pairs of routers with longer round-trip times, which may also contribute to longer delays in reacting to hot-potato routing changes. The "no peering" curve has some gaps that are 3 to 4 seconds long. These gaps are caused by the

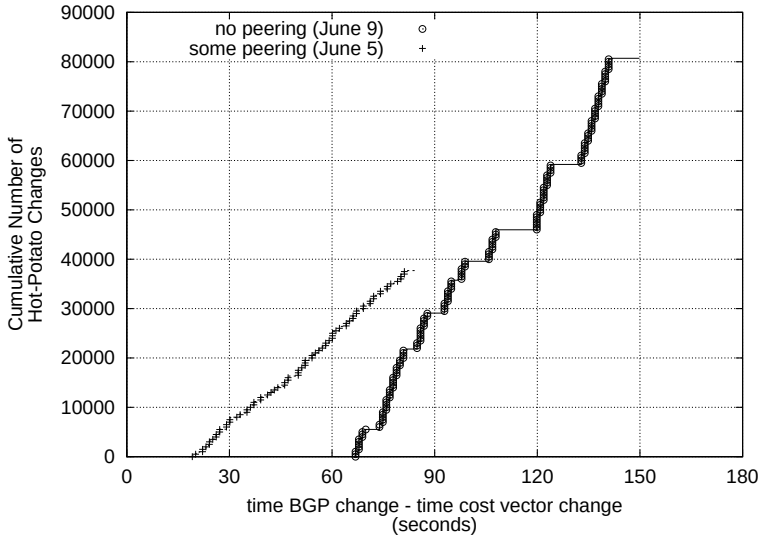


Figure 4.2: BGP transfers caused by one distance vector change.

minimum-route advertisement timer, which limits the rate of BGP updates in a session. The smaller steps (one second long) are due to the one-second granularity of the BGP monitor timestamp.

The transfer delay may also be responsible for the instances in Figure 4.1 in which the reaction time exceeds 80 seconds for the "first change" curve. These kinds of delays may be caused by the propagation of hot-potato BGP routing changes from one router to another, as shown in Figure 4.3. In the example, routers A and B are route reflectors and routers C, D, and E are egress points; C is a client of A, and D and E are clients of B. Initially, A and B select egress point D, with distances of 18 and 8, respectively. A is unaware of the route via E because B only advertises its best route to A. When the BD distance increases to 11:

1. The LSA is flooded throughout the network and each router computes new distances to D. For example, A and B compute new distances of 21 and 11, respectively.

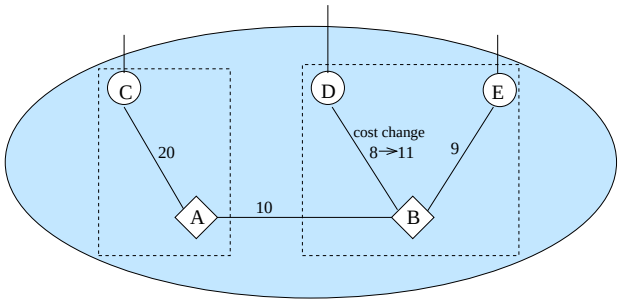


Figure 4.3: Router A waits for B's decision.

2. After their scan timers elapse, A and B rerun the BGP decision process. If A runs first, A selects the egress point C with a distance of 20, since this is smaller than 21. Sometime afterward, B selects egress point E.

3. B sends the new route (with egress point E) to A, and A selects egress point E with a distance of 19.

Suppose a distance vector change triggers a large number of BGP updates from B, but some of these updates do not trigger hot-potato changes in A. Then, A may have to wait for the transfer of a number of BGP updates before experiencing a hot-potato change. This behavior explains some of the reaction times longer than 80 seconds in Figure 4.1. Other instances with longer reaction times may also be due to false matches in associating a BGP routing change with a distance vector changes. Along the seven-month trace, each distance vector changes for which BGP takes more than 80 seconds to react trigger 807 BGP routing changes on average, whereas those that have smaller reaction times trigger 3398 BGP updates on average.

Combining the results of the "first change" curve in Figure 4.1 and the transfer delays in Figure 4.2, a router's reaction to distance vector changes may take 0–80 seconds for the first prefix and an additional 80 seconds (in extreme cases) for the remaining prefixes. Combining these effects, the vast majority of hot-potato changes take place within three minutes of the distance vector change, as shown by the "all changes"

curve in Figure 4.1.

4.1.3 Temporal and Spatial Variability

The influence of hot-potato routing varies significantly across time. Figure 4.4 presents the number[1] of hot-potato updates. For ease of presentation, the graph plots the days in increasing order of the number of hot-potato BGP routing changes and we only show the 46 days with higher number of hot-potato changes. The order of days in the x-axis are not the same across the three routers. The results show that on most days the routers did not experience *any* hot-potato routing changes. Still, on a few days the number was much higher. For the "no peering" router, one day had an unusually large number of hot-potato routing changes that were responsible for 82% of the BGP routing changes on that day. The variability across the days may stem from natural differences in the time and location of IGP weight changes and maintenance activity. The large variation across days makes it difficult to define a representative statistic for the frequency of hot-potato routing changes.

Comparing the three curves in Figure 4.4 highlights the influence of the location of the router on the likelihood of hot-potato routing changes. Over the period of our study, the "rich peering" router was always the least affected by distance changes, as seen by the bottom curve lying very close to the x-axis in Figure 4.4. The likelihood that a distance change affects the selection of the BGP best route depends on the proximity of the router to each of its nearby egress points. For the "rich peering" router, many of the prefixes have an egress point at the same PoP. Very few distance changes would cause the router to select a different egress point for these prefixes. This result suggests that a natural way to reduce the number of hot-potato routing changes would be to have rich peering at *every* PoP. However, having rich peering at all locations is unfeasible in practice, due to cost and geographic constraints. A service provider is bound to have

[1] Although the graph omits the values on the y-axis, the three curves are plotted in linear proportion starting at $y = 0$.

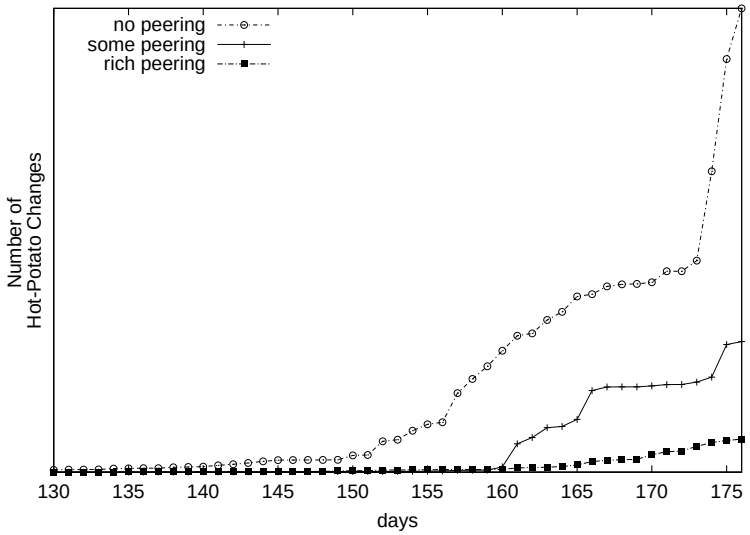


Figure 4.4: Hot-potato changes across days and locations.

routers in some remote locations that are not close to PoPs of the other large providers.

Here we are just observing hot-potato changes from three vantage points, the frequency of these changes "network-wide" are more often than the per-router results may indicate. There are hundreds of routers in the network and each of them experience a large number of hot-potato routing changes at different times. The impact of internal changes depends on both the location and the internal events that happened in a day. In the same day a router in one location of the network may experience a large number of hot-potato changes, whereas another has none.

4.1.4 Hot-Potato Variation Across Prefixes

Previous studies have shown that a small fraction of unstable prefixes are responsible for most of the BGP route updates [LAJ99, RWXZ02] and that the popular prefixes responsible for the bulk of the traffic have very few BGP updates [RWXZ02, ACBD04]. The BGP routes for the remaining prefixes stay the same for days or weeks at a time. An interesting question is to what extent these results translate to hot-potato

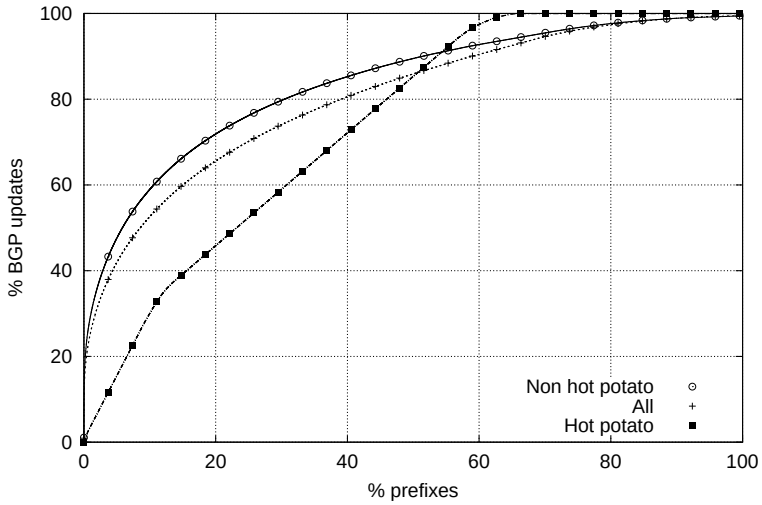


Figure 4.5: CDF of BGP updates across destination prefixes.

changes. Figure 4.5 plots the cumulative distribution of BGP update messages across the destination prefixes for the "no peering" router for June 2003. To compare our results with previous work, the graph plots the number of *BGP update messages* rather than the number of BGP routing changes. The prefixes are sorted according to their contribution to the number of BGP messages. The middle curve corresponds to all of the BGP messages. About 20% of the prefixes contribute 65% of the BGP updates, consistent with previous findings. However, the bottom curve shows that the distribution of BGP updates caused by *hot-potato* routing changes has a much more even spread across the prefixes.

The broader distribution across prefixes occurs because distance vector changes can affect the distances to reach the egress points for a wide variety of prefixes. Still, some prefixes do not experience *any* hot-potato BGP updates, as seen in the flat portion in the upper-right part of the graph. This part of the curve corresponds to prefixes with a very small number of egress points, including the prefixes that have a *single* egress point. Every router in the network would always pick this single egress

point as the best egress point for the prefix. Still, the relatively uniform distribution across the remaining prefixes may have important implications. For prefixes that generally have stable *eBGP*-learned routes, internal distance changes could be a primary cause of the BGP routing changes observed inside an AS. Since some of these prefixes may be responsible for a large volume of traffic, limiting the frequency of hot-potato routing changes may be useful to avoid large traffic shifts and transient performance disruptions. We study the impact of egress changes in the traffic matrix in Section 4.2.

4.1.5 Routing Shifts

Hot-potato routing can sometimes cause a router to change the egress points for multiple destination prefixes, which will lead to a large number of BGP update messages at the same time. Even if these destination prefixes carry no traffic, this burst of updates may disrupt the control plane by overloading and crashing routers. In Figure 4.6, we explore how many destination prefixes are affected at a single router when a distance change occurs. More than 99% of the distance changes do not affect the egress point for any prefix. The vast majority of intradomain events occur far away from the router, and as result do not affect the path distances for nearby egress points. Even when changes occur closer to the router, they might not affect the router's local ranking of the two closest egress points for a given prefix. However, when hot-potato routing changes *do* occur, the effects can be dramatic. For the "no peering" router in the top curve in Figure 4.6, 0.1% of the distance changes affect the BGP route for more than 40% of the prefixes.

These kinds of routing changes can lead to sudden increases in traffic at the new egress points and along the downstream paths. To estimate these effects we used Netflow measurements as explained in Section 3.3. In Figure 4.7, we replot the graph from Figure 4.6 for the timeframe for what we also have traffic data available. This plot shows that the percent of traffic that shifts is roughly the same or higher than the percent

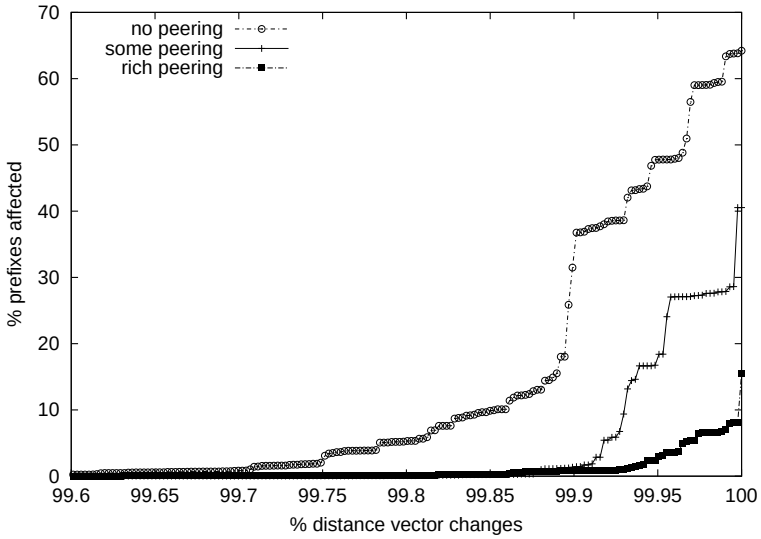


Figure 4.6: Fraction of prefixes affected by distance vector change.

of prefixes affected. This result gives a preliminary indication that some internal routing changes cause significant shifts in traffic. The next section studies the impact of routing changes in the traffic matrix in more detail.

4.2 Traffic Matrix Analysis

The design and operation of IP networks depends on a good understanding of the offered traffic. Although well-provisioned networks are designed to tolerate some fluctuation in the traffic matrix, large variations break the assumptions used in most designs. Fluctuations in traffic demands and changes in the prefix-to-egress mapping cause the traffic matrix to vary. We consider the natural question: *which of these two components are responsible for large variations in the traffic matrix?*

In this section, we explore the contributions of changes in the traffic demands $\mathcal{M}$ and prefix-to-egress mapping ε to the variations in the traffic matrix elements TM. Our analysis shows that, although most changes in ε have a small effect on the traffic

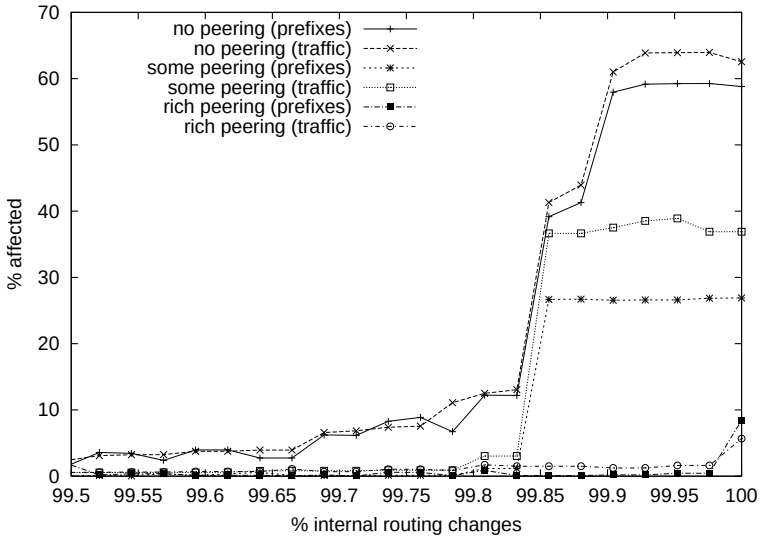


Figure 4.7: Fraction of traffic affected by internal changes.

matrix, many of the large variations in the traffic matrix are caused by changes in ε. Also, we show that, while most changes in ε are caused by external routing events, the small number of internal routing events that trigger hot-potato routing changes are more likely to cause larger shifts in traffic.

4.2.1 Definition of Traffic Variations

Figure 4.8 shows an example of how two traffic matrix elements (with the same ingress point i) change over the course of a day. The total traffic entering at the ingress point varies throughout the day, following a typical diurnal cycle. For the most part, the traffic $TM(i, e_1, t)$ has the same pattern, keeping the proportion of traffic destined to e_1 relatively constant. For most of the day, no traffic travels from ingress i to egress point e_2. The most significant change in the two traffic matrix elements occurs near the end of the graph. The traffic leaving via egress point e_1 suddenly decreases and, at the same time, traffic leaving via egress point e_2 increases. This shift occurred because a routing change caused most of the traffic with egress point e_1 to shift to egress point

e_2. The egress point e_2 also starts receiving traffic that had previously used other egress points (not shown in the graph), resulting in an increase for e_2 that exceeds the decrease for e_1. In the meantime, the total traffic entering the network at ingress i remained nearly constant.

The traffic experiences other relatively large downward spikes (labeled as "load variation"). These spikes may very well be associated with a routing change in another AS in the Internet that caused traffic to enter at a different PoP (this kind of traffic variation was called an "ingress-shift anomaly" in [LCD04]). In this thesis, we analyze traffic shifts caused by routing changes experienced by our network. Finding a signature of routing-induced traffic variations for one network is an important first step to infer other traffic variations that are caused by routing changes in other networks.

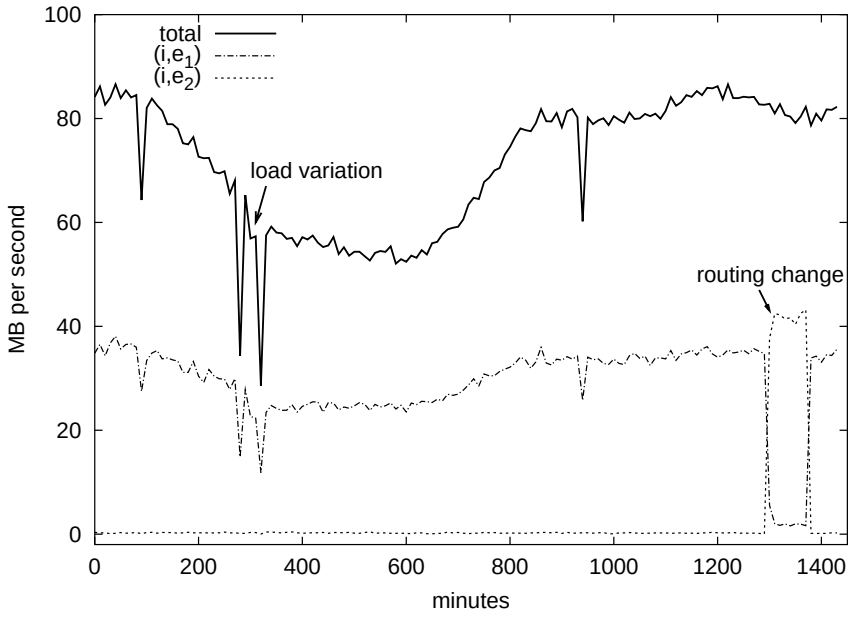


Figure 4.8: Sample traffic volume from one ingress to two egresses.

To analyze these kinds of traffic fluctuations, we define the variation of a traffic

matrix element at an interval t as:

$$\Delta TM(i, e, t) = TM(i, e, t) - TM(i, e, t - 1).$$

The next section decompose this variation to identify its cause: fluctuations in traffic demands or egress point shifts.

4.2.2 Changes in Traffic Demands vs. Egress Points

The variation of a traffic matrix element (ΔTM) is composed of the load variation (ΔL), which represents volume fluctuations on the traffic demands $\mathcal{M}$, and the routing shifts (ΔR), which accounts for changes in the prefix-to-egress mapping ε:

$$\Delta TM(i, e, t) = \Delta L(i, e, t) + \Delta R(i, e, t)$$

$\Delta L(i, e, t)$ represents the change in the volume of traffic for all destination prefixes that did *not* change their egress point from the previous time interval (i.e., $\varepsilon(i, p, t) = \varepsilon(i, p, t - 1) = e$):

$$\Delta L(i, e, t) = \sum_{\substack{p \in P: \\ \varepsilon(i,p,t) = e \\ \varepsilon(i,p,t-1) = e}} \mathcal{M}(i, p, t) - \mathcal{M}(i, p, t - 1)$$

Fluctuations in the traffic demands may occur for a variety of reasons, such as changes in user or application behavior, adaptations caused by end-to-end congestion control, or even routing changes in other domains.

The routing variation $\Delta R(i, e, t)$ considers the destination prefixes that shifted *to* egress point e during time interval t or shifted *from* e to another egress point in t:

$$\Delta R(i, e, t) = \sum_{\substack{p \in P: \\ \varepsilon(i,p,t) = e \\ \varepsilon(i,p,t-1) \neq e}} \mathcal{M}(i, p, t) - \sum_{\substack{p \in P: \\ \varepsilon(i,p,t) \neq e \\ \varepsilon(i,p,t-1) = e}} \mathcal{M}(i, p, t - 1)$$

Note that if a routing change occurs within the time interval t, we associate *all* of the traffic associated with that prefix in that time interval with the new egress point.

Not all traffic matrix elements carry the same volume of traffic, and the volume of traffic from an ingress to an egress PoP varies over time. How do we judge if a change in the traffic is "large"? There is no absolute standard: one approach might be to judge the size of the change in traffic matrix element relative to the average traffic for that element. However, this is not useful here, because the traffic process itself is non-stationary. It has daily and weekly cycles, as well as level shifts resulting from routing changes. The *relative* change $\Delta TM(i,e,t)/TM(i,e,t)$ (or $\Delta TM(i,e,t)/\max(TM(i,e,t),TM(i,e,t-1))$) seems appealing. However, this metric places too much emphasis on large relative changes to small values; for example, a traffic matrix element with 1 kbit/sec might easily experience a 50% relative change in traffic without having any significant effect on the network. An alternative metric would be the *absolute* change $\Delta TM(i,e,t)$. However, a shift of (say) 10 MB/sec may be significant for one ingress point but not for another. Another option would be to normalize the value of $\Delta TM(i,e,t)$ by the total traffic entering ingress point i at time t, which would capture changes in the *fraction* of the incoming traffic that uses a particular egress point. However, this metric depends on the "current" traffic demand at ingress i (which could be low at certain times) and may not accurately reflect the strain imposed on the network by the traffic change. Another extreme approach would be to consider the capacity of the network, and define as large any traffic shift that causes a link to be overloaded. Besides being difficult to compute, this metric is too closely tied to the current design of the network, and is not useful for most typical applications of the traffic matrix such as capacity planning or anomaly detection. Instead, we want a metric that captures properties of the traffic matrix itself, such as how large the traffic changes are relative to the normal variations of traffic matrix elements.

For that, we should consider what type of process we observe, namely, a differ-

ence process. Over short time periods, we can approximate the traffic with a linear process $y_t = \alpha + \beta t + x_t$, where x_t is a zero-mean stochastic process with variance σ^2. We observe the differences $\Delta y_t = y_t - y_{t-1}$, which will form a *stationary* process, with mean β and variance $2\sigma^2$. Thus we can approximate the difference process by a stationary process, and measure deviations from the mean relative to the standard deviation of this process. We measure $2\sigma(i,e)^2$ on the traffic variation process $\Delta L(i,e,\cdot)$ (using the standard statistical estimator), and use this to normalize the traffic variations, i.e., we then observe $\Delta T\tilde{M}(i,e,t) = \Delta TM(i,e,t)/\sqrt{2}\sigma(i,e)$, $\Delta\tilde{L}(i,e,t) = \Delta L(i,e,t)/\sqrt{2}\sigma(i,e)$, and $\Delta\tilde{R}(i,e,t) = \Delta R(i,e,t)/\sqrt{2}\sigma(i,e)$.

If the variance of the process x_t was time dependent, it might make sense to use a moving average to estimate the process variance at each point in time, i.e. $\sigma(i,e,t)^2$, and use this to normalize the traffic variations. We tried such an approach, but it made little difference to the results, and so we use the simpler approach described above.

Figure 4.9 presents a scatter plot of $\Delta T\tilde{M}(i,e,t)$ versus $\Delta\tilde{R}(i,e,t)$ for all the valid measurement intervals t. The high density of points close to zero shows that large traffic variations are not very frequent (99.88% of the traffic variations are in the $[-4,4]$ range). Points along the horizontal line with $\Delta\tilde{R}(i,e,t) = 0$ correspond to traffic variations that are not caused by routing changes, whereas points along the diagonal line correspond to variations caused almost exclusively by routing changes. Points in the middle are caused by a mixture of routing changes and load variation. Figure 4.9 shows that both load and routing are responsible for some large variations. Routing changes, however, are responsible for the *largest* traffic shifts. Indeed, one egress-point change made a traffic matrix element vary more than 70 times the standard deviation.

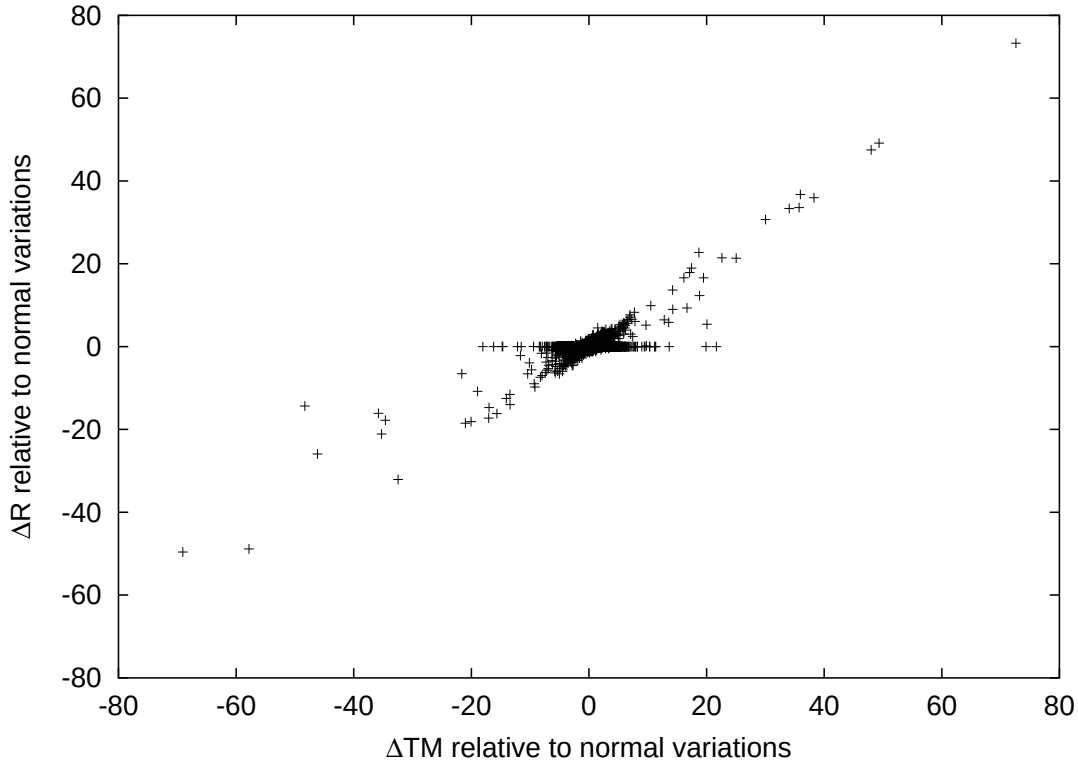


Figure 4.9: Scatter plot of $\Delta\tilde{TM}$ versus $\Delta\tilde{R}$ for all traffic matrix elements over the seven-month period.

4.2.3 Internal vs. External Routing Changes

The prefix-to-egress mapping ε may change because of either internal or external routing events. *External routing changes* represent any changes in the set of egress points that an AS uses to reach a destination prefix. For example, in Figure 3.7, the neighbor AS might withdraw the route for p_2 from the router e, resulting in a change in ε. External routing changes may be caused by a variety of events, such as an internal routing change in another domain, a modification to the local BGP routing policy, or a failure at the edge of the network. In contrast, *internal routing changes* stem from changes in the routing inside the AS, due to equipment failures, planned maintenance,

or traffic engineering. These events affect the prefix-to-egress mapping because of hot-potato routing.

Figure 4.10 shows the cumulative distribution functions of $\Delta\tilde{R}$ caused by hot-potato routing and by external BGP changes. For comparison, we also present the CDF of a normal distribution, which is drawn from randomly generated Gaussian data with standard deviation equal 1. Although the routing events are rare (only 0.66% of non-zero $\Delta\tilde{TM}$ are caused by eBGP changes and 0.1% by hot-potato changes), this result shows that there are significant cases where these events are large, to very large. In particular, approximately 5% of traffic shifts caused by hot-potato routing are at least one order of magnitude larger than normal variations. As discussed in Section 4.1.4, a single internal change is more likely to affect a large number of destination prefixes, including the popular destinations receiving large amounts of traffic.

By analyzing the source of traffic variation for individual traffic matrix elements, we see that the likelihood of changes in the prefix-to-egress mappings can vary significantly from one ingress router to another. Figures 4.11 and 4.12 present the same data as in Figure 4.9 for two sample traffic matrix elements (note that the axis are different across the two graphs). Some traffic matrix elements have no traffic variation caused by routing changes (Figure 4.11), whereas others have few very large egress shifts (Figure 4.12). We computed the percent of the traffic matrix elements (i, e) that have large to very large traffic shifts. We define large as more than 4 times the normal traffic variations for (i, e) and very large more than 10 times. Approximately 25% of elements (i, e) in our study have no large traffic variation, and the vast majority of them (85.7%) have no very large traffic variation. The differences across the traffic matrix elements have two main explanations:

- **Size of traffic matrix element**. Some traffic matrix elements carry little traffic. Because of hot-potato routing, most of the traffic from an ingress router exits the network at few egress PoPs. For instance, most of the traffic entering in San Diego

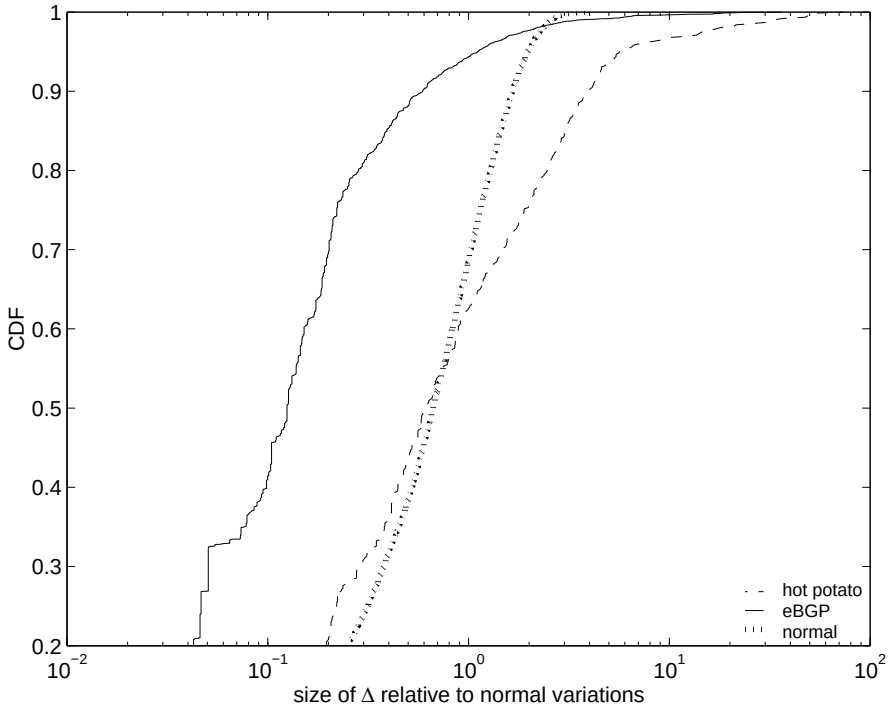


Figure 4.10: Cumulative distribution function of $\Delta\tilde{R}$ caused by hot-potato routing and eBGP.

is likely to stay in the west coast. Therefore, the traffic element San Diego to New York carries very little traffic at any time.

- **Impact of internal events**. The likelihood of hot-potato routing changes varies significantly from one ingress point to the other (as shown in Section 4.1.3), depending on the location in the network and the proximity to the various egress points. For our eight ingress points, the fraction of BGP routing changes caused by internal events varies from 1% to 40%. As a result, the likelihood of large traffic shifts caused by hot-potato routing varies significantly from one traffic matrix element to another.

Out of the traffic matrix elements that do experience large traffic variations, 15% have an average of more than one large traffic variation per week. The small

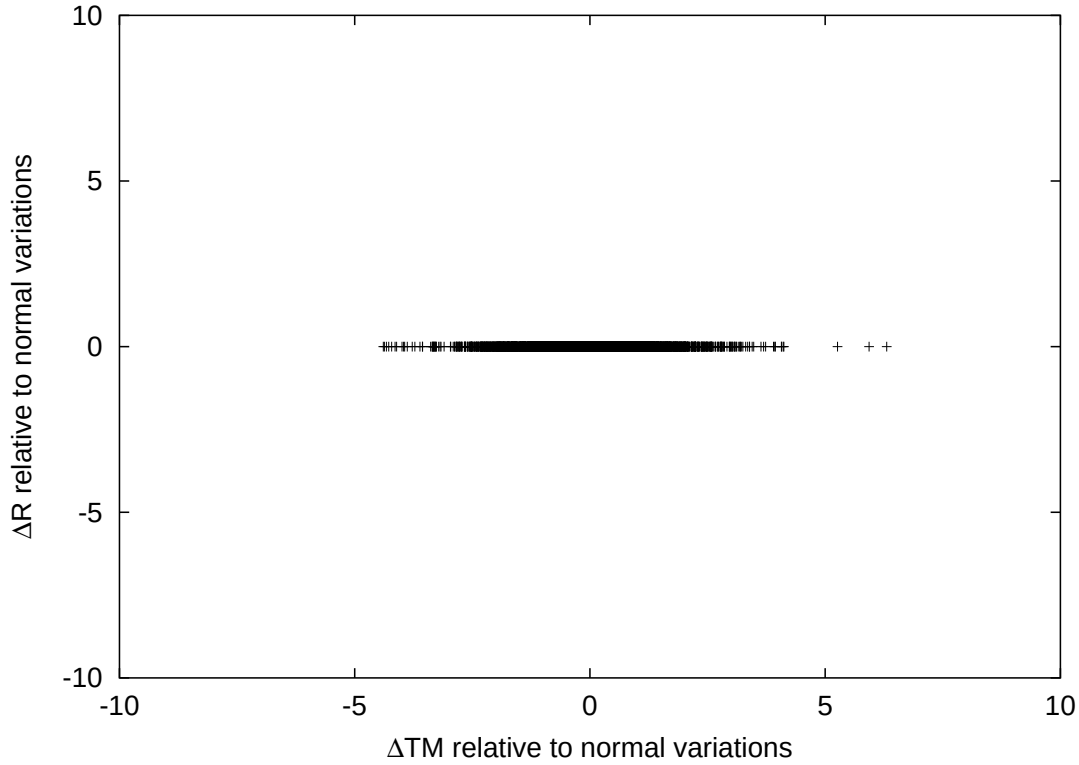


Figure 4.11: Scatter plot of $\Delta\tilde{TM}$ versus $\Delta\tilde{R}$ for a traffic matrix element that has no routing-induced traffic variations over the seven-month period.

percentage of elements that experience large traffic variations combined with the low frequency large shifts per element may lead to the incorrect conclusion that these events are irrelevant. However, if we consider the network-wide frequency of large traffic shifts, these events happen fairly often. To show this, we have counted the number of 10-minute measurement intervals for which at least one of our eight vantage points experienced a large traffic variation. On average, the network experiences a large traffic variation every four and half hours. Large traffic variations caused by routing changes happen every 2.3 days, and very large routing-induced traffic variations happen every 5.9 days. If our analysis considered all of the PoPs in the network, the overall frequency

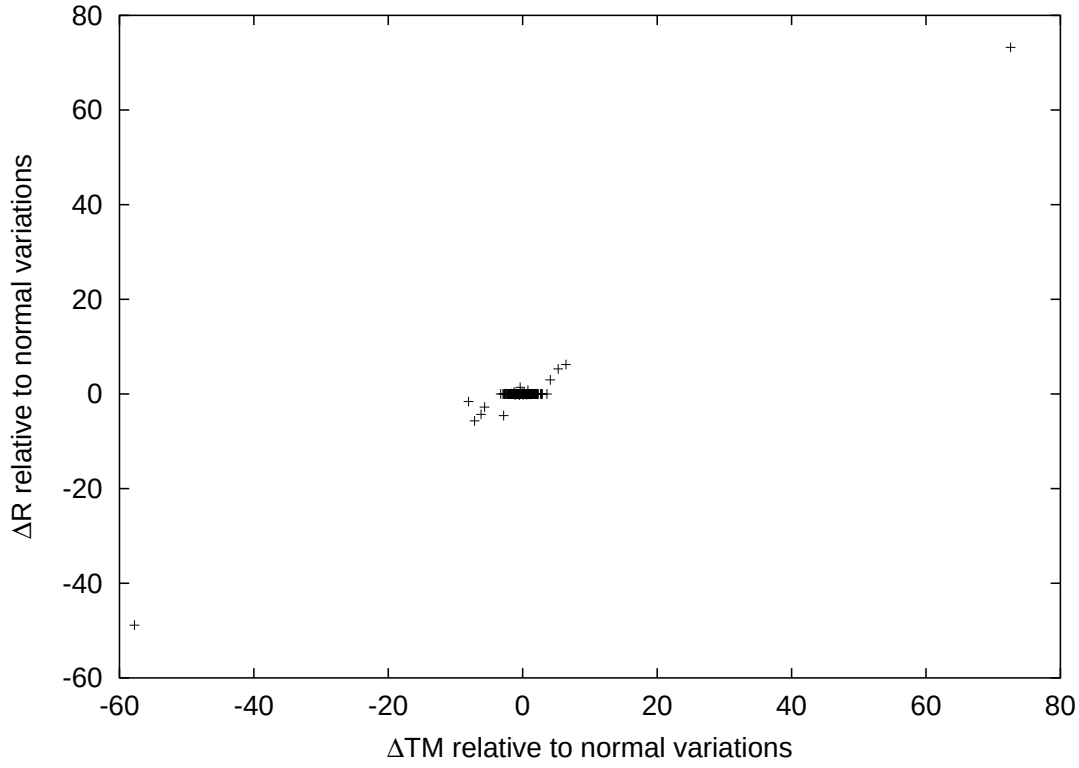


Figure 4.12: Scatter plot of $\Delta\tilde{TM}$ versus $\Delta\tilde{R}$ for a traffic matrix element that has few very large routing-induced traffic shifts over the seven-month period. One traffic shift was over 70 times normal traffic variations!

of large traffic variations would be even higher.

4.3 Implications of Hot Potatoes

Hot-potato routing changes impact IP networks performance and robustness. The previous sections show that hot-potato routing can cause large number of egress shifts and the largest variations in the traffic matrix. Hot-potato changes in BGP routing also influence network performance by causing shifts in the flow of traffic to neighboring domains and extra delays in the convergence of the forwarding plane. In addition, hot-

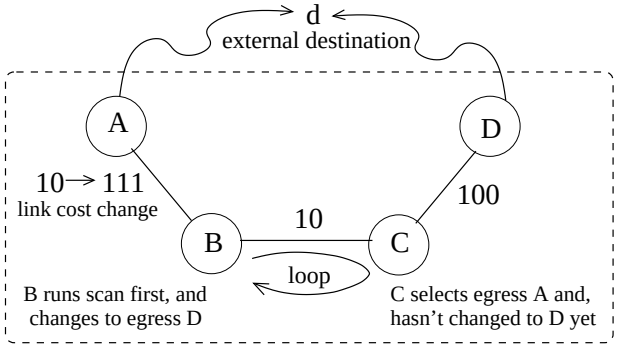


Figure 4.13: Transient forwarding loop for packets destined to d.

potato changes can introduce inaccuracy in active measurements of the forwarding plane and external monitoring of BGP update messages.

4.3.1 Slow Forwarding-Plane Convergence

Compared to other kinds of routing changes, hot-potato routing changes cause longer delays in forwarding-plane convergence, since each router must recompute its IGP routes *and* rerun the BGP decision process before updating the forwarding table. Differences in when the routers revisit their BGP decisions can lead to transient forwarding loops, as illustrated in Figure 4.13. In this example, the AS has four routers and two egress points to prefix d. The numbers on the edges represent the IGP link weights, and we omit the full-mesh of iBGP sessions for simplicity. At first, routers B and C both identify router A as the closest egress point, causing C to direct traffic to d through B. When the weight of the B–A link increases to 111, both routers eventually switch to the route learned at D. However, if B runs its BGP decision process first and updates its forwarding table, B starts forwarding traffic destined to d toward D while C continues to forward the traffic toward A—resulting in a forwarding loop.

During the interval before C runs its decision process and updates its forwarding-table entry for d, all packets destined to d are caught in a forwarding loop between B and C. The packets would repeatedly traverse the loop until the IP Time-

to-Live (TTL) field expires, causing one of the routers to discard the packet. The forwarding loop causes packet loss for the hosts communicating with d, and increased congestion for other traffic traversing the BC link. Depending on the alignment of the BGP scan timers on the two routers, this problem can persist for up to 60 seconds, even though the intradomain routing protocol has converged. If TCP transfer latency or the iBGP hierarchy cause larger delays in forwarding-plane convergence, the loops can persist even longer.

Note that the extra convergence delay for hot-potato routing changes does *not* affect the stability of the forwarding path for the iBGP sessions themselves. The IP packets sent over iBGP sessions travel between routers within the backbone and the forwarding of traffic between these routers depends only on the IGP. The delivery of BGP updates to our route monitor is not affected either, since the network has a single egress point to reach the monitor.

According to a previous study of packet-level measurements in a large ISP backbone [HMMD02], most forwarding loops last for less than 10 seconds. This result is consistent with typical delays for IGP convergence [InCM$^+$02, AJY00]. However, the study also found that, for one of the links, about 35% of the loops persisted for 10–60 seconds. Based on our results, we speculate that these forwarding loops can be explained by hot-potato routing changes.

4.3.2 Measurement Inaccuracies

Active measurement analysis of the performance of IP networks or passive measurement analysis of routing and traffic that ignore the interaction between IGPs and BGP may lead to inaccurate conclusions.

Active Probes of the Forwarding Plane

The effects of slow forwarding-plane convergence may be difficult to capture using traditional active measurement techniques. Service providers and third-party measurement companies deploy probe machines in various parts of the network to exercise the paths between pairs of hosts. Referring to Figure 4.13, suppose the provider connected one probe machine to router A and another to router D. Probe packets sent from A to D would traverse the path A–B–C–D. When the IGP weight of the BA link changes, these probes might experience temporary loss while the IGP reconverges. However, the forwarding path of the probe packets would *not* be affected by the 60-second scan timer since there would be no change in the egress point used to reach the destination address of the probe packets; both B and C continue to use the egress point D to reach the destination probe machine. This is true, in general, for probe machines that connect to a single location inside an AS. As such, measurements between these kinds of probe machines would only capture the transient effects of *IGP* convergence, and not the combined *IGP-BGP* convergence process. Accurately capturing the performance impact of hot-potato routing changes would require a more complex active measurement infrastructure with probe machines reachable through multiple egress points.

External Analysis of BGP Updates

A hot-potato routing change does not necessarily cause an AS to advertise new BGP routes to neighboring ASes. First, the export policy for the eBGP session might filter the route. This decision depends on the commercial relationship with the neighbor (e.g., a route learned from one peer would not be exported to another) and on whether route aggregation is performed. Second, the router might decline to forward the new route if it does not differ significantly from the old route. For example, routers typically perform *non-transitive attribute filtering* to avoid propagating routes that differ only in

local attributes (like BGP next-hop or local-preference) rather than global ones (such as AS path). Third, the router might not propagate the route due to BGP timers, such as the minimum-route advertisement timer, that are used to pace the rate of updates to neighboring ASes. If the router changes its best BGP route for the prefix multiple times during the interval, the intermediate BGP routes would not be visible to the neighbor.

For a rough estimate of the externally-visible updates, we look at BGP routing changes that affect the *AS path attribute*, since these would be propagated to neighboring domains subject to the export policy and the BGP timers. In Figure 4.14, if A is using the route via AS 3, when C switches to egress point B; we would not classify this routing change as externally visible. However, if router A's best route was the one learned from AS 2, the AS path would change; router C would propagate the new route to its eBGP neighbors. Looking over the month of June, we estimate that around 14% of the hot-potato routing changes seen at the "no peering" router would be sent to its neighbors; this would account for 5% of the externally-visible BGP routing changes. For the "some peering" router, these two numbers are 5% and 2%, respectively—about 60% smaller than for the "no peering" router. Although these average numbers are relatively small, the values vary substantially from day to day; on one day *all* hot-potato updates at all three routers had changes in the AS path.

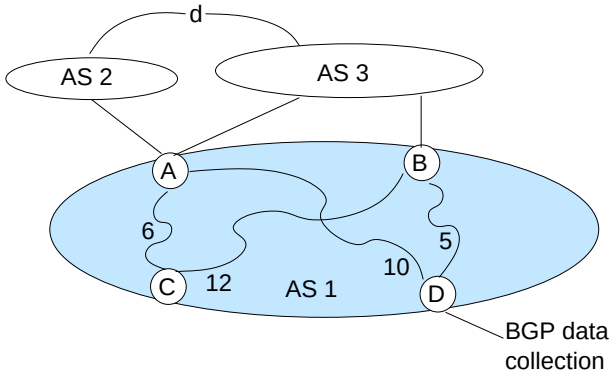


Figure 4.14: BGP changes are not detected at data collection point.

These externally-visible BGP updates may affect the results of research studies based on public BGP routing data [RV, RIP] collected from eBGP sessions with routers in large ASes throughout the Internet. Depending on which router in an ISP network connects to these public servers, the contribution of hot-potato routing changes to the data may vary significantly! Hot-potato routing implies that different routers in an AS may pick different BGP-level routes. Referring again to Figure 4.14, suppose that router A chooses the route via AS 2 based on an arbitrary tie break, such as the router ID. Based on hot-potato routing, router D selects the route through B and router C selects the route through A. As such, BGP data collected from D would only reveal the route via AS 3. Now suppose that a failure occurs on the link connecting router A to AS 2. Then, both A and C would switch to the route via AS 3, which may lead to a change in the properties of the end-to-end paths for traffic entering AS 1 at router C. However, the link failure does *not* cause a change in the BGP route at D and, as such, the change is not visible to the measurement system. When viewed from outside the AS, a hot-potato routing change that affects a large number of prefixes in one network may also be indistinguishable from a BGP session reset at another nearby location. This lack of visibility is a challenge for tools for locating the origin of BGP instability such as [CSK03, FMM+04].

Analysis of Traffic Effected by Egress Changes

The results in Section 4.2 show that the likelihood of changes in the prefix-to-egress mappings can vary significantly from one ingress router to another. In particular, we have shown in Section 4.1.3 that some ingress points may be much more susceptible to hot-potato routing changes than others, making analysis of routing stability very dependent on where the data are collected. For example, the study in [RWXZ02] showed that popular destination prefixes do not experience BGP routing changes for days or weeks at a time. In addition to studying RouteViews and RIPE BGP feeds, the analy-

sis included iBGP data from two of the eight routers used in our current study. In our analysis, these two routers did not experience many hot-potato routing changes. Had the analysis in [RWXZ02] analyzed a router that experiences several hot-potato routing changes a day, the conclusions might have been quite different. In fact, hot-potato routing changes can affect a large number of prefixes (see Sections 4.1.5 and 4.1.4), both popular and not, so we might reasonably expect popular destinations to experience changes in their egress points. A preliminary analysis across all eight vantage points confirms that popular destination prefixes have more BGP instabilities from vantage points that experience more hot-potato routing changes.

The analysis in Section 4.2 shows that large changes in the traffic matrix elements occur relatively infrequently. In addition, most changes in the prefix-to-egress mapping do not lead to large traffic shifts, consistent with the results in [ACBD04]. Yet, these two results do *not* imply that routing changes are not a significant contributor to large changes in the traffic matrix elements. In fact, the opposite is the case. A small number of routing changes are indeed responsible for a relatively large fraction of the (few) large traffic shifts. In addition, long traces are necessary to draw conclusions about infrequent (yet significant) events. The study in [ACBD04] draws on five traces of 6–22 hours in duration, making it difficult to conclude definitively if large traffic shifts occur and whether routing contributes to them.

Ignoring Egress Changes in Traffic Matrix Analysis

Previous work on measuring and analyzing traffic matrices has assumed that routing is stable, in part because fine-grained routing data is sometimes difficult to collect. Most of the work on traffic matrix *estimation* [CDWY00, MTSD02, ZRLD03] assumes that there are no changes in the prefix-to-egress mapping or the intradomain paths between the ingress and egress points. Even work on direct *measurement* of the traffic demands [LPC+04, FGL+01] has used only daily routing snapshots, although

the work in [ACBD04] is a notable exception. Using out-of-date routing information runs the risk of associating some traffic measurements with the wrong elements in the traffic matrix. In some cases, the routing changes might lead to second-order effects on the traffic (e.g., by causing congestion or increasing the round-trip time) that may appear in the data, but the primary affect of the traffic *moving* to a different egress point is obscured—as is the *reason* for the variation in the traffic. In addition, changes in the prefix-to-egress mapping may cause large fluctuations in *multiple traffic matrix elements at the same time*, which would be obscured if the traffic matrix is computed or analyzed without regard for routing changes.

4.4 Summary

Our results suggest that hot-potato routing may play an important role in BGP routing changes, and that BGP updates can lag 60 seconds (or more!) behind the related IGP events. This delayed interaction can lead to surprisingly long delays in forwarding-plane convergence that greatly exceed the typical delays for IGP convergence [InCM+02, AJY00]. The frequency and impact of hot-potato routing depends on the topology and configuration of the network under study. Indeed, even routers in the same network can be more or less impacted by hot-potato routing changes depending on their location and on the intradomain routing changes that happened during the measurement period. This dependency on vantage point has important implications for network performance and routing measurements. We also show that large traffic variations, while rare, do sometimes happen. Although most routing changes typically do not affect much traffic, routing is usually a major contributor to large traffic variations. In particular, internal routing changes are responsible for the largest shifts in the traffic matrix because of hot-potato routing.

In this chapter, we have done an empirical study of hot-potato routing changes. The impact of such egress changes varies significantly across time and router location.

Our measurement methodology is useful for diagnosing the cause of routing changes and our results show the importance of this egress selection policy. The next chapter presents a model that network administrators can use to study network sensitivity to internal routing changes.

Chapter 5

Network Sensitivity Analysis

In the previous chapter, we have used measurements of one ISP network to show that hot-potato routing can have a significant impact on BGP route selection and on the traffic matrix. For example, one IGP routing change was measured to change 62% of a router's BGP table. We call such abrupt BGP route shifts caused by IGP distance changes *hot-potato disruptions*. Motivated by these findings, this chapter proposes metrics that capture the impact of intradomain changes on the control and data planes.

The rest of this chapter is organized as follows. First, we motivate the need for network sensitivity metrics and informally define the control plane as a routing matrix and the data plane as a traffic matrix. We discuss the challenges of modeling hot-potato routing and our assumptions in Section 5.2. In Section 5.3, we model the routing matrix while focusing on a single destination prefix. We then extend the model in Section 5.4 to capture control plane and data plane sensitivity. Section 5.5 presents our methodology to compute control plane sensitivity in real networks and analyzes control plane sensitivity of a tier-1 AS to link and router failures.

5.1 Motivation

Even though understanding a network's sensitivity to a change is a crucial step in designing a robust network, current traffic engineering techniques and tools for network design and management have limited capability to analyze this sensitivity. In fact, there is even little terminology to describe and measure network sensitivity. In this thesis, we address this problem by defining metrics for characterizing both control plane and data plane sensitivity to routing changes inside an AS.

We model the control plane as an *interdomain routing matrix* RM that maps a node and a destination to a set of potential egress points. A node has a set of potential egress points for each destination, because we do not consider the tie-break after the comparison of IGP distances. The routing matrix is a function of the routes sent to an AS and the locally implemented routing policies. The routing matrix stores the set of egress points that are best according to BGP *and* that have equal IGP distances from the node. Changes in both IGP and BGP may affect a network routing matrix. We express this dependence informally in the following relation, where $\oplus$ represents the combination of information from the two protocols:

$$RM \leftarrow IGP \oplus BGP$$

As discussed in Section 4.2, the data plane is often modeled as a *traffic matrix* (TM). We adopt this abstraction because, when combined with the routing matrix, it is easier to determine metrics such as the overall network load or individual link loads. The traffic matrix is computed from the combination of the traffic demands $\mathcal{M}$ and the routing matrix. Thus, changes in traffic demands or routing decisions may result in changes in the traffic matrix. We represent this combination as $\otimes$ in the following relation.

$$TM \leftarrow RM \otimes \mathcal{M}$$

Informally, we can now talk about measuring robustness, or put another way, the sensitivity of a network (or elements of a network) to perturbations. We are interested in the *control plane sensitivity* to an IGP change ΔIGP. We can capture this as

$$\Delta RM \leftarrow \Delta IGP \oplus BGP$$

and say that the network is robust if for all "small" ΔIGP we produce a "small" ΔRM. This naturally extends to a quantification of *data plane sensitivity*, which captures perturbations to the traffic matrix

$$\Delta TM \leftarrow \Delta RM \otimes \mathcal{M}$$

and can be derived from the control plane perturbations.

Note that control plane and data plane sensitivities are not necessarily correlated. For example, a small routing change can produce a large shift in traffic, and a large routing change can leave traffic unchanged.

5.2 Modeling Hot-Potato Routing

This section first outlines the challenges of modeling the routing behavior and the impact of internal routing changes on the selection of external routes and consequently on traffic. Then, we present the assumptions we make to model hot-potato routing.

5.2.1 Challenges

Modeling the sensitivity of routing and traffic to intradomain routing changes in an ISP network requires a good understanding of how the network uses IGP and BGP for routing, the resulting routing matrix, the traffic demands, and how the traffic demands and routing matrix combine to generate the traffic matrix. Below we outline the main challenges of modeling this complex system:

- **IGP**: Each AS has the freedom to design its internal routing infrastructure and select the appropriate IGP to route packets within the network. For scalability reasons, both OSPF and IS-IS allow hierarchical routing by dividing a network into areas [Moy98, Cal90]. When areas are employed, packets are forwarded along shortest paths within an area, but may deviate from the shortest paths when they traverse multiple areas. As a result, a model of hot-potato routing needs to incorporate the link weights and the division of routers into areas and accurately model the IGP distance computation.

- **BGP**: Modeling the complete set of egress points per prefix within an ISP can become complicated due to the iBGP architecture of a network. The BGP standard requires a full mesh of iBGP connections between all BGP speaking routers in the network [RL95]. This ensures that all the routers know about the best routes learned by every other router via eBGP. To overcome the resulting scalability issues, networks with a large number of BGP speakers usually structure them into some form of iBGP hierarchy; "router-reflectors" and "confederations" are two popular ways of forming this hierarchy [HM01]. An unfortunate side-effect is that some BGP routes are available to only a subset of BGP routers inside a network. This can lead to scenarios where different routers in a network have different sets of egress points.

- **Routing matrix**: After we have an abstract model of both IGP and BGP decisions, we still need to combine them to form the routing matrix, which depends on the interaction of the two protocols. Some parts of this interaction are standardized such as steps 0 through 6 of the BGP decision process, whereas some like the tie-breaking rule after the comparison of IGP distances are not. This implies that the interaction between IGPs and BGP depends on router implementation or specific network configuration.

- **Traffic demands and traffic matrix**: The traffic demand placed on a network depends on the traffic sent and received by the end-users attached to customer and peer networks, and the locations in which they connect to the network. Consequently, traffic demands vary considerably over time. In general, it is hard to obtain a representative snapshot of the traffic demands and the resulting traffic matrix. Even though measurement and estimation of the traffic matrix have received considerable research attention of late [FGL+01, MTSD02, ZRLD03], measurements are aggregated or sampled and not all routers in the network are capable of collecting the required data [ZRLD03].

5.2.2 Assumptions of the Model

To reduce the complexity of modeling all these details, we make the following simplifying assumptions:

1. **All routers know all routes to reach a destination prefix**. As mentioned earlier, this assumption is valid when BGP speakers are organized in a full iBGP mesh. In ASes that use an iBGP hierarchy this assumption no longer holds. However, even in the case of an iBGP hierarchy, the following condition is enough for our model to capture the network sensitivity correctly: every router at the very least learns the route that it would have picked as the best route had it learned all the eBGP-learned routes for a given prefix.

2. **Well-configured iBGP**. Griffin and Wilfong [GW02b] describe scenarios in which iBGP misconfigurations can lead to route deflections. This means that routers in the forwarding path to an egress point for a prefix disagree on the selection of the egress point leading to deflections and loops in an extreme case. We assume that the network configuration satisfies the conditions specified in [GW02b] for avoiding such loops and deflections.

3. **Stable BGP routing snapshots**. We assume that BGP policies and eBGP routes are stable. Our model works with snapshots of BGP routes and analyzes the impact of intradomain routing changes on snapshot of routes.

4. **Stable snapshots of traffic demands**. Similarly, our model takes as input snapshots of the traffic demands to analyze the impact of intradomain routing changes on traffic.

These assumptions allow us to model the state of routing system and snapshots of traffic. Our model does not capture the routing dynamics or traffic load variations directly. A single underlying event may cause a series of routing messages, and routers in the network may take some time to converge to a consistent state. We do not model the sequence of messages exchanged to reach a stable routing state. Instead, we model a routing change from one stable state to another stable state reached after the convergence phase.

5.3 Regions and Region Shifts

This section first introduces concepts from multidimensional data analysis that inspired us in our network sensitivity analysis. Then, it describes a graph model and the concepts of graph regions and region shifts that together form the basis of our network model of hot-potato routing. The set of definitions presented here capture the interaction between BGP and IGP when considering a single destination prefix. In Section 5.4, we build on this terminology to define control plane and data plane sensitivity metrics. Table 5.1 summarizes the notation introduced in this section and the following one, and used in the remainder of the chapter.

Basic	
Universe of vertices	$\mathcal{U}$
Undirected weight graph	$G = (N, L, w, d), N \subset \mathcal{U}$
Egress set	$E \subset \mathcal{U}$
v's rank for egress node e	$d(v, e), v \in N$ and $e \in E$
Region of an egress node e	$R_e(G, E) \subseteq N$
Region index set of a vertex v	$RI(G, E, v) \subseteq E$
Class of topology changes	ΔG
Probability function for topology changes	$\mathbb{P}(\Delta G)$
Regional	
Region-shift function	$\mathcal{H}(G, E, v, \delta)$
Vertex sensitivity	$\eta(G, E, \Delta G, \mathbb{P}, v)$
Impact of a topology change	$\theta(G, E, \delta)$
Graph sensitivity	$\sigma(G, E, \Delta G, \mathbb{P})$
Control Plane Sensitivity	
Set of destination prefixes	P
Mapping of prefixes to egress sets	$\mathcal{E} : P \rightarrow 2^{\mathcal{U}}$
Routing-shift function	$\mathcal{H}^{RM}(G, P, v, \delta)$
Node routing sensitivity	$\eta^{RM}(G, P, \Delta G, \mathbb{P}, v)$
Routing impact of a topology change	$\theta^{RM}(G, P, \delta)$
Control plane sensitivity	$\sigma^{RM}(G, P, \Delta G, \mathbb{P})$
Data Plane Sensitivity	
Set of ingress nodes	$\mathcal{I}$
Ingress-to-prefix traffic matrix	$\mathcal{M}(v, p), v \in N$ and $p \in P$
Total traffic volume entering v	$\mathbb{T}(v)$
Traffic-shift function	$\mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta)$
Ingress node traffic sensitivity	$\eta^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}, v)$
Traffic impact of a topology change	$\theta^{TM}(G, P, \mathcal{M}, \delta)$
Data plane sensitivity	$\sigma^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M})$

Table 5.1: Summary of definitions and metrics.

5.3.1 Multidimensional Data Analysis

Extracting useful information from a network configuration is a very difficult task. We are interested in the sensitivity of a network to hot-potato disruptions, which depends on a number of factors including network topology, routing policies, location in the network, the specific network changes considered, etc. To make matters worse, there seems to be no "one size fits all" metric that can capture the routing sensitivity in a meaningful way. Our approach to managing this complexity is inspired by the area of databases where similar challenges arise — *Online Analytic Processing* (OLAP) [GBLP97, CD97, GHQ95]. The key to OLAP design is to arrange data in a multidimensional cube, and then provide natural ways to "slice and dice" the data along multiple dimensions.

For instance, imagine a three-dimensional OLAP data cube in which each cell contains sales totals indexed by product sold, city of sales, and day of sales. It is possible to explore the sales data by dicing it into one, two, or three-dimensional slices, and then aggregating and displaying a slice in various ways. For example, if we fix a product and city, then we obtain a one-dimensional slice of the cube that captures the sales over time of the chosen product in the chosen city. This slice could be used to compute an average over all days, or to find the days of maximum sales. In a similar way, if we fix a city, we can obtain a two-dimensional slice of the cube that captures the sales of all products in that city over all days. An interesting characteristic of most data cubes is that the dimensional data has some natural hierarchical structure. For example, we could generate a new data cube by "rolling-up" the time dimension to the level of *month*, the location dimension to the level of *country*, and the product dimension to the level of *category*.

Figure 5.1 presents the OLAP data cube we define for the purpose of exploring the impact of hot-potato routing on a network. Our dimensions are *location* (routers), *network change* (representing a fixed class of changes such as "all single link failures"

or "all single node failures"), and IP prefixes. We find that a large number of interesting hot-potato sensitivity queries can be computed with a *very simple data cube* — each data cell contains only a single bit! This bit is set if the associated router changes egress points for the associated prefix when the associated network change is applied. In this thesis, we will not explore the fine-grained hierarchy of our dimensional data (for example, routers are naturally grouped into PoPs, which can be grouped into regions, which in turn can be grouped into countries), but the utility of this should be clear.

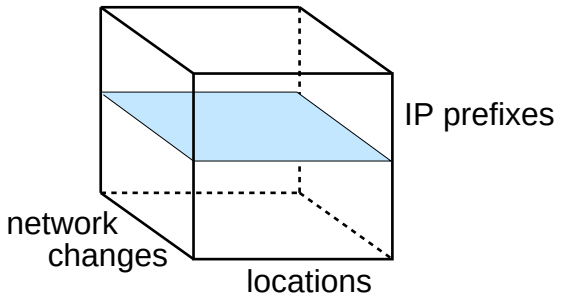


Figure 5.1: Data cube for network sensitivity analysis.

In this section, we study slices of the data cube that correspond to an IP prefix such as the one represented by the shaded surface in Figure 5.1. We define metrics of region-shift sensitivity computed over this surface.

5.3.2 Definitions

Let $\mathcal{U}$ be the universe set of vertices. Given an undirected weighted graph $G = (N, L, w, d)$, where $N \subset \mathcal{U}$, and a non-empty set of vertices $E \subset \mathcal{U}$, we say that E is the set of *egress vertices*.[1] These egress vertices represent the set of egress points for a destination prefix.

Every vertex $v \in \mathcal{U}$ has a local ranking of egress vertices. The function $d(v, e)$ returns v's rank for egress vertex e. When either v or $e \notin N$, $d(v, e) = \infty$. If $d(v, e) <$

[1] We consider N a subset of a greater universe of vertices because we want to study cases in which vertices are deleted or added to the graph G.

$d(v, e')$, then v is *closer* to e than to e'. The ranking function d represents the IGP distance between routers in a network, and consequently its computation depends on specific details of the intradomain routing protocol and the network configuration. In a network with no IGP hierarchy, $d(v, e)$ is the sum of the weights of the edges in a shortest path between v and e. If the network has a more complex IGP structure, then $d(v, e)$ can be computed using a model of the IGP hierarchy [FGL+00, SG04].

Given a graph G with ranking function d and a egress set E, each egress vertex $e \in E$ induces a *region* $R \subseteq N$ such that:

$$R_e(G, E) = \{v | \forall v \in N \text{ and } d(v, e) \leq d(v, e'), \forall e' \in E, e \neq e'\}.$$

This construction divides G into $|E|$ regions, and each region $R_e(G, E)$ is a shortest distance tree rooted in e. Note, though, that the division of G into regions is not a partition of the vertices in G because regions are not necessarily mutually disjoint. For instance, if $d(v, e) = d(v, e')$ for $e, e' \in E$, then $v \in R_e(G, E) \cap R_{e'}(G, E)$. We allow vertices to be in more than one region to model the potential of the tie-break decision (after the hot-potato step in the BGP decision process) to use any of the associated regions. Note that if $e \notin N$, then $R_e(G, E) = \emptyset$.

We define $RI(G, E, v) \subseteq E$ as the *region index set* of a vertex v in a graph G divided into regions according to the egress set E.

$$RI(G, E, v) = \{e | \forall e \in E, v \in R_e(G, E)\}.$$

The region index set of a vertex v is the set of egress nodes that are closest to v. If E is the egress set for a destination prefix p, then $RI(G, E, v)$ represents v's entry in the routing matrix when considering p. If v is disconnected from the graph, then $\forall e \in E, d(v, e) = \infty$ and $RI(G, E, v) = \emptyset$.

For example, the graph G in Figure 5.2 presents the internal topology of the AS shown in Figure 1.1. The figure shows the division of this topology into regions for the destination prefix p from our previous example. The egress set is the egress set for p, so $E = \{A, B\}$. All other nodes are internal nodes. An arrow from vertex i to vertex j represents that vertex j is the successor of vertex i in the shortest path to a egress e. The distances from C to A and B are 9 and 10, respectively; therefore, C is in the region of A (i.e., it forwards packets using egress point A). The region $R_A = \{A, C, D, F, J\}$ represents all routers that forward packets to p using A as the egress point, and the region $R_B = \{G, J\}$ similarly represents the routers that use B as the egress. Note that router J is in both R_A and R_B, hence $RI(G, E, J) = \{A, B\}$.

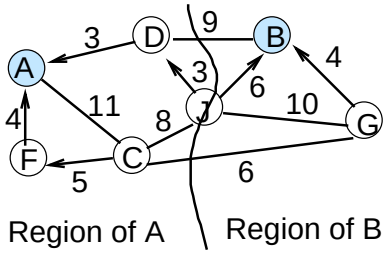


Figure 5.2: Example of the division of G into regions.

A *topology change* of G is a function $\delta : (N, L, w, d) \rightarrow (N', L', w', d)$ that deletes one or more edges or vertices, changes the weight of one or more edges, or adds one or more edges or vertices. Let ΔG be the class of topology changes to be applied to G, such as the set of single edge deletions. Every topology change δ has a probability $\mathbb{P}(\delta)$ associated with it such that $\sum_{\delta \in \Delta G} \mathbb{P}(\delta) = 1$.

For each $\delta \in \Delta G$ and $v \in N$, we can compute the *region-shift function* $\mathcal{H}(G, E, v, \delta)$ as follows:

$$\mathcal{H}(G, E, v, \delta) = \begin{cases} 1, & \text{if } RI(G, E, v) \neq RI(\delta(G), E, v) \\ 0, & \text{otherwise} \end{cases}$$

Intuitively, the function $\mathcal{H}$ determines whether a vertex experiences an egress change, i.e., whether a vertex v changes regions after a topology change δ is applied to G, thereby shifting region boundaries. Note that RI is a set of egress vertices, hence the removal or addition of an element is considered a region shift. When an egress $e \in E$ is deleted from G, it no longer belongs to any region and consequently the value of the region-shift function for e is one. For all other possible types of topology changes δ, a egress vertex e cannot change regions, therefore $\mathcal{H}(G, E, e, \delta) = 0$ for all $e \in E$.

Let us re-examine the example presented in Figure 5.2. Suppose that ΔG is the set of single edge deletions. Now consider the scenario where $\delta =$ *delete edge* CF. In an IP network, the deletion of an edge represents a link failure. Figure 5.3 presents the resulting graph $\delta(G)$ with the new division into regions. The distance of C to A changed from 9 to 11 while the distance of C to B remained 10. Hence, C changes from R_A to R_B and $\mathcal{H}(G, E, C, \delta) = 1$. All the other vertices are not affected by the change, so $\forall v \in N, v \neq C, \mathcal{H}(G, E, v, \delta) = 0$.

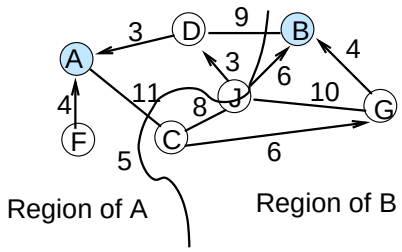


Figure 5.3: Example of the division of vertices into regions after deleting the edge CF.

5.3.3 Regional Sensitivity

To analyze a graph's sensitivity to topology changes with respect to a set of egress vertices, we construct a two-dimensional surface where there is a point for each pair (v, δ). The value of each point (v, δ) is set to $\mathcal{H}(G, E, v, \delta)$. Then, we introduce metrics that capture the sensitivity of a vertex to region shifts and the impact of a topol-

ogy change on all the vertices in the graph by computing averages and maxima over one-dimensional slices of this plane as represented in Figure 5.4.

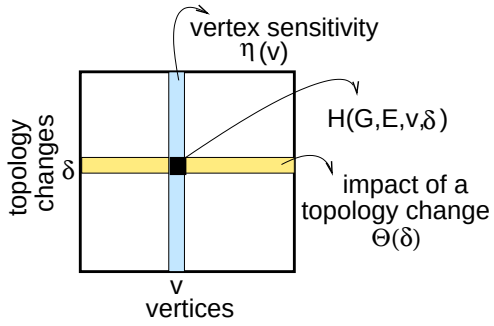


Figure 5.4: Regional sensitivity metrics are computed over a two-dimensional surface of vertices and topology changes.

Vertex Sensitivity

The *vertex sensitivity* ($\eta \in [0, 1]$) describes the expected regional sensitivity of a vertex $v \in N$ given the set of topology changes ΔG with probability $\mathbb{P}$.

$$\eta(G, E, \Delta G, \mathbb{P}, v) \quad = \quad \sum_{\delta \in \Delta G} \mathcal{H}(G, E, v, \delta).\mathbb{P}(\delta)$$

Vertex sensitivity captures the likelihood of a vertex to change regions when applying the class of topology changes ΔG in G. In this section, we discuss sensitivity for a given set of parameters for a graph G, class of topology changes ΔG, and topology change probability function $\mathbb{P}$. For conciseness of notation, we omit these parameters, hence $\eta(G, E, \Delta G, \mathbb{P}, v) \equiv \eta(E, v)$. We also define the average vertex sensitivity ($\hat{\eta}$), which is useful for evaluating the sensitivity of a graph.

$$\hat{\eta}(G, E, \Delta G, \mathbb{P}) \quad = \quad \frac{1}{|N|}. \sum_{v \in N} \eta(G, E, \Delta G, \mathbb{P}, v)$$

Consider the graph G presented in Figure 5.5. Assume that all edges have the same weight, $E_1 = \{A, B\}$, ΔG is the class of single edge deletions, and all $\delta \in \Delta G$ are independent and have equal probabilities. In this scenario, the four vertices are divided into two regions as illustrated in the figure.

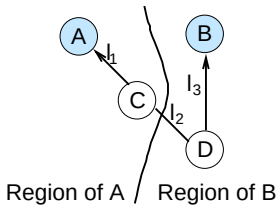


Figure 5.5: Example of a graph with vertices divided into regions as induced by $E_1 = \{A, B\}$.

Table 5.2 presents a cross-tabulation over the values of $\mathcal{H}(E_1, v, \delta)$ for all vertices in Figure 5.5. Each column δ_i corresponds to the deletion of the edge l_i. The last column presents the vertex sensitivity $\eta(E_1, v)$ for each vertex. Vertices A and B are egresses, therefore no edge deletion can cause them to change regions and both $\eta(E_1, A)$ and $\eta(E_1, B)$ equal 0. Out of the three possible topology change, only the deletion of edge l_1 causes vertex C to change from R_A to R_B. Hence $\eta(E_1, C) = \frac{1}{3}$. The computation of D's sensitivity is similar. We will describe the last row of the table when we define the impact of a topology change and graph sensitivity.

In general, values of $\eta(E, v)$ vary from 0 to 1. When $\eta(E, v)$ is close to 0, v is not sensitive to the class of topology changes ΔG, i.e., it is unlikely that v changes regions given the division induced by E. By definition, $\eta(E, e) = 0$ for any egress $e \in E$ when considering any class of topology changes that does not include the deletion of egress vertices. Routers that are in the set of egress points for a destination prefix always prefer routes learned from eBGP. Therefore, no intradomain changes (excluding the router crashing) can have an impact on route selection for such routers.

	δ_1	δ_2	δ_3	$\eta(E_1, v)$
A	0	0	0	0
B	0	0	0	0
C	1	0	0	$\frac{1}{3}$
D	0	0	1	$\frac{1}{3}$
$\theta(E_1, \delta_i)$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\sigma(E_1) = \frac{1}{6}$

Table 5.2: Sensitivity metrics using egress set E_1 (Figure 5.5.

A value of $\eta(E, v) = 1$ means that v is sensitive to any topology change $\delta \in \Delta G$. Consider the graph presented in Figure 5.5 with only B as egress ($E_2 = \{B\}$). Figure 5.6 shows the division in regions for this new scenario and Table 5.3 shows the values of the sensitivity metrics. In this example, A just has one route to reach the destination prefix p, thus any edge deletion disconnects A from p. Since the deletion of any edge in the graph disconnects A from the egress B, $\eta(E_2, A) = 1$.

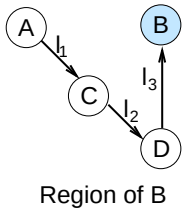


Figure 5.6: Example of a graph with egress set $E_2 = \{B\}$.

	δ_1	δ_2	δ_3	$\eta(E_2, v)$
A	1	1	1	1
B	0	0	0	0
C	0	1	1	$\frac{2}{3}$
D	0	0	1	$\frac{1}{3}$
$\theta(E_2, \delta_i)$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{3}{4}$	$\sigma(E_1) = \frac{1}{2}$

Table 5.3: Sensitivity metrics using egress set E_2 (Figure 5.6.

Impact of a Topology Change

The *impact of a topology change* ($\theta \in [0,1]$) determines the fraction of the vertices in N that experience region shifts after a given topology change.

$$\theta(G,E,\delta) \quad = \quad \frac{1}{|N|}.\sum_{v \in N} \mathcal{H}(G,E,v,\delta)$$

We define a metric that captures the expected impact ($\hat{\theta}$) of the class of topology changes ΔG with probability function $\mathbb{P}$:

$$\hat{\theta}(G,E,\Delta G,\mathbb{P}) \quad = \quad \sum_{\delta \in \Delta G} \theta(G,E,\delta).\mathbb{P}(\delta)$$

Let us revisit the example in Figure 5.5 to analyze the impact of edge deletions. The last row of Table 5.2 presents the values $\theta(E_1,\delta)$ for each edge deletion. The deletion of edge l_2 does not cause any vertex to change regions. This topology change has the least impact. Both the deletion of l_1 and l_3 impact one of the vertices in the graph, thus these changes have a higher value of θ. The last row of Table 5.3 presents $\theta(E_2,\delta)$ for the scenario depicted in Figure 5.6. When only B is in the egress set, the deletion of l_3 causes all other vertices to change regions, making $\theta(E_2,\delta_3) = \frac{3}{4}$.

As with vertex sensitivity, for any egress set E and topology change δ, values of $\theta(E,\delta)$ vary between 0 and 1. $\theta(E,\delta) = 0$ represents a change that causes no region shifts in the graph, whereas $\theta(E,\delta) = 1$ happens when δ is a change that causes all the vertices in the graph to change regions. When the egress set is composed of all vertices in the graph (i.e., $E = N$), then no topology change (with the exception of vertex deletions) causes a vertex to change regions. This configuration is not sensitive to changes in the graph and $\theta(E,\delta) = 0$. Note, however, that this scenario is extremely unlikely in an IP network. First, peering is not available everywhere. Second, the set

of egress points for a prefix is usually much smaller than the number of routers in the network. On the other hand, if δ is the deletion of all $e \in E$, then all vertices $v \in N$ change regions and $\theta(E, \delta) = 1$. This other extreme is also unlikely for prefixes with more than one egress point.

Graph Sensitivity

Both the average vertex sensitivity $\hat{\eta}$ and the expected impact of a topology change $\hat{\theta}$ allow us to compare the overall region-shift sensitivity for $(G, E, \Delta G, \mathbb{P})$.

Lemma 5.3.1

$$\hat{\eta}(G, E, \Delta G, \mathbb{P}) = \hat{\theta}(G, E, \Delta G, \mathbb{P})$$

Proof :

$$\begin{aligned}
\hat{\eta}(G, E, \Delta G, \mathbb{P}) &= \frac{1}{|N|} \cdot \sum_{v \in N} \eta(G, E, \Delta G, \mathbb{P}, v) \\
&= \frac{1}{|N|} \cdot \sum_{v \in N} \sum_{\delta \in \Delta G} \mathcal{H}(G, E, v, \delta).\mathbb{P}(\delta) \\
&= \sum_{\delta \in \Delta G} \frac{1}{|N|} \cdot \sum_{v \in N} \mathcal{H}(G, E, v, \delta).\mathbb{P}(\delta) \\
&= \sum_{\delta \in \Delta G} \theta(G, E, \delta).\mathbb{P}(\delta) \\
&= \hat{\theta}(G, E, \Delta G, \mathbb{P})
\end{aligned}$$

□

Therefore, we define *graph sensitivity* as

$$\begin{aligned}
\sigma(G, E, \Delta G, \mathbb{P}) &= \hat{\eta}(G, E, \Delta G, \mathbb{P}) \\
&= \hat{\theta}(G, E, \Delta G, \mathbb{P}).
\end{aligned}$$

We can now compute the graph sensitivity for the example presented in Figure 5.5. Over all the twelve possible v, δ combinations only two experience region

changes and they have the same probability, so $\sigma(E_1) = \frac{1}{6}$. The scenario presented in Figure 5.6 is more likely to have region changes after an edge deletion. The graph sensitivity metric reflects this increased sensitivity. For E_2, $\sigma(E_2) = \frac{1}{2}$ and $\sigma(E_2) > \sigma(E_1)$.

When $\sigma(E) = 0$, the graph G with egress set E is extremely robust to the topology changes defined in ΔG. This happens when $E = N$ and ΔG does not include vertex deletions. Values of σ close to one arise when any topology change in ΔG causes all vertices to change regions. The only scenario in which σ is one is when ΔG contains only one change that is the deletion of all the egresses together. In general, we expect the graph sensitivity to be $0 < \sigma(E) < 1$.

5.4 Network Sensitivity Metrics

We now use the terminology introduced in the previous section to model hot-potato routing. Then, we present metrics that capture both control and data sensitivity to IGP routing changes.

5.4.1 From Regions to Hot Potatoes

We model an AS as a graph G, where vertices represent routers, edges are IP-level links, edge weights are the IGP distance associated with a link interface, and the IGP distances (computed with or without IGP hierarchy) are incorporated by the ranking function d. The set of egress points for a destination prefix p is represented as $\mathcal{E}(p)$. We assume that all routers $e \in \mathcal{E}(p)$ announce equally-good routes to reach p. This assumption implies that our model considers the set of best BGP routes after applying policies. In other words, we are only considering routes that are tied through step 5 in the BGP decision process (see Table 2.1).

A region R_e is the set of routers that use the route announced by egress point e to forward traffic to destination prefix p. A hot-potato routing change is an egress point change resulting from one or more intradomain distance changes caused by some

underlying event (such as a fiber cut, interface down, router crash, IGP weight change for maintenance, etc.). We model intradomain routing changes as a topology change. In this scenario, the region-shift function $\mathcal{H}$ determines whether a router experiences an egress change for a destination prefix when a particular intradomain routing change happens.

One can think of vertex sensitivity as a metric that, given a network and a set of intradomain routing changes, determines the likelihood of a router to change its selection of egress point for a particular prefix. The impact of a topology change can be used to determine the fraction of routers that experience an egress change after an intradomain routing change. The graph sensitivity metric σ reflects the overall impact of internal routing changes on the egress point that routers choose to forward packets to a destination prefix.

By the definition of regions, when a router is equidistant from two egress points, we consider it to be in two different regions at the same time. In practice, a router breaks this tie using a configuration-specific rule. For instance, by default Cisco routers prefer the older route [Cisa], which leads to different decisions depending on the order of routing changes. A model that captures this tie-breaking rule would have to simulate routing dynamics. Even though there are some deterministic options for breaking this tie (such as comparing the ID of the egress points) that could potentially be incorporated into our model, the choice of the tie break is particular for each network configuration.

Instead, a network should be robust independent of routing dynamics (otherwise, the network design would be counting on luck!). The definition of region-shift function introduced in the previous section captures the worst-case sensitivity, because it considers that if there is any chance that a vertex v might change regions, then $\mathcal{H}(G, E, v, \delta) = 1$. We refer to this definition of region-shift function as $\mathcal{H}_{worst}$.

$$\mathcal{H}_{worst}(G, E, v, \delta) = \begin{cases} 1, & \text{if } RI(G, E, v) \neq RI(\delta(G), E, v) \\ 0, & \text{otherwise} \end{cases}$$

Because of the tie-breaking rule, when a node is in more than one region at the same time, some topology changes may change the region index RI and yet cause no egress change in practice. We define $\mathcal{H}_{best}$ as the best case scenario for the region-shift function. If there is any egress vertex that belongs to $RI(G, E, v)$ and $RI(\delta(G), E, v)$, then we consider that v does not have an egress change.

$$\mathcal{H}_{best}(G, E, v, \delta) = \begin{cases} 1, & \text{if } RI(G, E, v) \cap RI(\delta(G), E, v) = \emptyset \\ 0, & \text{otherwise} \end{cases}$$

By computing the sensitivity metrics using $\mathcal{H}_{worst}$, we determine an upper bound for regional sensitivity. Similarly, using $\mathcal{H}_{best}$ determines a lower bound. In practice, the sensitivity metrics should lie between these two values depending on the tie-breaking rule adopted and the order of routing changes.

For clarity of notation, we define control plane and data plane sensitivity metrics using the region-shift function $\mathcal{H}$ instead of defining upper and lower bounds based on $\mathcal{H}_{worst}$ and $\mathcal{H}_{best}$. Our case study of the ISP network in Section 5.5 uses these definitions to compute upper and lower bounds of network sensitivity.

5.4.2 Control Plane Sensitivity

So far, our model has focused on the analysis of sensitivity when considering only one destination prefix. However, hot-potato disruptions result from a large number of routes changing simultaneously because of an intradomain routing change. Thus, we add another dimension to the data cube to capture the set of destination prefixes P and analyze the network state as a three-dimensional data cube (as the one presented

in Figure 5.1) to understand the impact of a topology change on all the prefixes in the routing matrix.

Each destination prefix p has a set of egress points $\mathcal{E}(p)$. The set of egress points, however, is not unique per prefix. It is often the case that a number of destination prefixes share the same set of egress points. For example, the two networks presented in Figure 1.1 peer into two locations (peering routers A and B). In this example, all destination prefixes of customers of the neighbor AS share the egress set $\{A, B\}$. Let P be a set of destination prefixes, and $\mathcal{E}$ the mapping of prefixes to egress sets ($\mathcal{E} : P \rightarrow 2^{\mathcal{U}}$.

The *routing-shift function* ($\mathcal{H}^{RM}(G, P, v, \delta)$) represents the fraction of v's BGP table that change egress points after the topology change δ. This metric is analogous to the one empirically measured in Figure 4.6.

$$\mathcal{H}^{RM}(G, P, v, \delta) = \frac{1}{|P|} . \sum_{p \in P} \mathcal{H}(G, \mathcal{E}(p), v, \delta)$$

Routing sensitivity metrics are computed similarly to the region-shift metrics by replacing the region-shift function $\mathcal{H}$ with the routing-shift function $\mathcal{H}^{RM}$. One can view the routing-shift function as a transformation of the one-dimensional slice of the data cube for a pair (v, δ). After applying the transformation $\mathcal{H}^{RM}(G, P, v, \delta)$ for every pair (v, δ), we obtain a two-dimensional projection that is equivalent to the plane presented in Figure 5.4.

Node Routing Sensitivity

The *node routing sensitivity* ($\eta^{RM} \in [0, 1]$) describes a router's expected fraction of route shifts under the set of topology changes ΔG.

$$\begin{aligned}\eta^{RM}(G,P,\Delta G,\mathbb{P},v) &= \sum_{\delta\in\Delta G}\mathcal{H}^{RM}(G,P,v,\delta).\mathbb{P}(\delta)\\ \hat{\eta}^{RM}(G,P,\Delta G,\mathbb{P}) &= \frac{1}{|N|}.\sum_{v\in N}\eta^{RM}(G,P,\Delta G,\mathbb{P},v)\end{aligned}$$

Node routing sensitivity is useful for determining the routers in the network that are more susceptible to hot-potato changes. This metric does not differentiate between a router that experiences a few route shifts for most $\delta \in \Delta G$ and another that experiences rare but large route shifts. To differentiate between the two cases, we introduce η^{RM}_{max} to represent the worst case route shift for each node $v \in N$.

$$\eta^{RM}_{max}(G,P,\Delta G,v) = \max_{\delta\in\Delta G}\mathcal{H}^{RM}(G,P,v,\delta)$$

Node routing sensitivity η^{RM} can also be represented as the average vertex sensitivity across all prefixes as demonstrated by the following lemma.

Lemma 5.4.1

$$\eta^{RM}(G,P,\Delta G,\mathbb{P},v) = \frac{1}{|P|}.\sum_{p\in P}\eta(G,\mathcal{E}(p),\Delta G,\mathbb{P},v)$$

Proof :

$$\begin{aligned}\eta^{RM}(G,P,\Delta G,\mathbb{P},v) &= \sum_{\delta\in\Delta G}\mathcal{H}^{RM}(G,P,v,\delta).\mathbb{P}(\delta)\\ &= \sum_{\delta\in\Delta G}\frac{1}{|P|}.\sum_{p\in P}\mathcal{H}(G,\mathcal{E}(p),v,\delta).\mathbb{P}(\delta)\\ &= \frac{1}{|P|}.\sum_{p\in P}\sum_{\delta\in\Delta G}\mathcal{H}(G,\mathcal{E}(p),v,\delta).\mathbb{P}(\delta)\\ &= \frac{1}{|P|}.\sum_{p\in P}\eta(G,\mathcal{E}(p),\Delta G,\mathbb{P},v)\end{aligned}$$

□

Consider again the graph presented in Figure 5.5. Assume that the number of prefixes $|P| = 10$ and that six prefixes use egress set E_1 (Figure 5.5) and remaining four use egress set E_2 (Figure 5.6). Table 5.4 presents the node routing sensitivity $\eta^{RM}(v)$ for each vertex in G computed using the values $\eta(E_1, v)$ and $\eta(E_2, v)$ presented in Tables 5.2 and 5.3, respectively. Since B is in both E_1 and E_2, no single edge deletion causes it to change regions with respect to E_1 or E_2. Hence, it is the least sensitive vertex in G; in fact, $\eta^{RM}_{max}(B) = 0$. Vertex A changes regions after any edge deletion when the egress set is E_2, but never changes regions when the egress set is E_1. Hence, we compute $\eta^{RM}(A) = \frac{0\times6+1\times4}{10} = 0.4$. Even though by examining η^{RM} vertex D is less sensitive than A and B, η^{RM}_{max} indicates that there is at least one topology change (in this case δ_3) for which all its routes change egresses.

	$\eta^{RM}(v)$	$\eta^{RM}_{max}(v)$
A	0.4	0.6
B	0	0
C	0.47	0.6
D	0.33	1

Table 5.4: Node routing sensitivity for the graph presented in Figure 5.5 with egress sets E_1 and E_2.

Routing Impact of a Topology Change

The *routing impact of a topology change* ($\theta^{RM} \in [0, 1]$) represents the average across all routers of the fraction of each router's BGP table that change egresses after an intradomain routing change.

$$\theta^{RM}(G, P, \delta) = \frac{1}{|N|} \cdot \sum_{v \in N} \mathcal{H}^{RM}(G, P, v, \delta)$$

$$\hat{\theta}^{RM}(G, P, \Delta G, \mathbb{P}) = \sum_{\delta \in \Delta G} \theta^{RM}(G, P, \delta).\mathbb{P}(\delta)$$

The routing impact of a topology change represents the average number of entries in the routing matrix that are changed because of a topology change. Similar to node routing sensitivity, we define a metric to represent the node that is most impacted by each topology change (θ^{RM}_{max}).

$$\theta^{RM}_{max}(G, P, \delta) \quad = \quad \max_{v \in N} \mathcal{H}^{RM}(G, P, v, \delta)$$

As with node routing sensitivity, the routing impact of a topology change can also be defined in terms of the impact of a topology change on each egress set equivalence class.

Lemma 5.4.2

$$\theta^{RM}(G, P, \delta) = \frac{1}{|P|} . \sum_{p \in P} \theta(G, \mathcal{E}(p), \delta)$$

We omit the proof because the steps are similar to that of Lemma 5.4.1.

Table 5.5 shows the routing impact of the deletion of each edge in G. The deletion of edge l_3 is the change with greatest impact. Even though it only impacts one vertex when considering egress set E_1, it impacts all vertices for E_2. Indeed, we see from Table 5.5 that both $\theta^{RM}(\delta_3)$ and $\theta^{RM}_{max}(\delta_3)$ are higher than the impact of δ_1 and δ_2.

	$\theta^{RM}(\delta)$	$\theta^{RM}_{max}(\delta)$
δ_1	0.25	0.6
δ_2	0.2	0.4
δ_3	0.45	1

Table 5.5: Routing impact of single edge deletions on the graph presented in Figure 5.5 with egress sets E_1 and E_2.

Overall Control Plane Sensitivity

The *overall control plane sensitivity* ($\sigma^{RM} \in [0, 1]$) is the average node routing sensitivity or the expected routing impact of a class of topology changes. Control plane sensitivity represents the average fraction of the routing matrix that shifts in response to internal perturbations.

$$\begin{aligned} \sigma^{RM}(G, P, \Delta G, \mathbb{P}) &= \hat{\eta}^{RM}(G, P, \Delta G, \mathbb{P}) \\ &= \hat{\theta}^{RM}(G, P, \Delta G, \mathbb{P}) \end{aligned}$$

The overall control plane sensitivity for the example in Figure 5.5 with egress sets E_1 and E_2 is $\sigma^{RM}(G, P, \Delta G, \mathbb{P}) = 0.3$. Overall control plane sensitivity provides an aggregated view of network sensitivity to IGP changes and can be used as one metric of comparing the robustness of the control plane of different network designs.

Perhaps more useful for determining the robustness of a network are the worst case node routing sensitivity (η^{RM}_{max}) and routing impact of topology changes (θ^{RM}_{max}). High values of η^{RM}_{max} mean that there is at least one router in the network that has a high probability of experiencing hot-potato disruptions, which may then lead to router overload. Similarly, topology changes with a high routing impact may lead to the overload of the control plane. A robust network should minimize σ^{RM}_{max}.

$$\begin{aligned} \sigma^{RM}_{max}(G, P, \Delta G, \mathbb{P}) &= \max_{v \in N} \eta^{RM}_{max}(G, P, \Delta G, v) \\ &= \max_{\delta \in \Delta G} \theta^{RM}_{max}(G, P, \delta) \end{aligned}$$

By identifying the most disruptive topology changes, network administrators can plan for them before maintenance activities or, longer term, add extra links or routers to reduce or avoid the most disruptive events. Practical constraints may prevent a network

from being free of hot-potato disruptions. Knowledge of the areas of the network that are most vulnerable to hot-potato disruptions can be used when selecting the location to connect customers. For instance, it may be economically more advantageous to connect customers that use interactive applications such as voice and gaming in locations that are less susceptible to disruptions.

5.4.3 Data Plane Sensitivity

The previous section presented metrics to study variations in the routing matrix caused by IGP changes, i.e., the impact of hot-potato routing on the control plane. In this section, we combine routing with traffic demands and introduce metrics that measure the impact of hot-potato routing changes on the ingress-to-egress traffic matrix.

Let P be the set of destination prefixes and $\mathcal{I} \subseteq N$ be the set of ingress routers. $\mathcal{M}$ is an $|\mathcal{I}| \times |P|$ matrix, representing the ingress point to destination prefix traffic demand matrix. An element (v, p) of $\mathcal{M}$ represents the volume of traffic from an ingress router $v \in \mathcal{I}$ to a destination prefix $p \in P$. In contrast with Chapter 4, this chapter uses snapshots of the traffic demands and egress sets. Therefore, we omit the reference to time in the notation $\mathcal{M}(v, p, t)$. We redefine the data cube from Figure 5.1 to study data plane sensitivity. Each cell of the data cube presented in (v, δ, p) now contains the value $\mathcal{H}(G, \mathcal{E}(p), v, \delta).\mathcal{M}(v, p)$, which represents the volume of traffic from a node v to a prefix p if v changes regions when δ is applied to G. Given the ingress-to-prefix demand matrix, the total inbound traffic at an ingress node v ($\mathbb{T}(v)$) is:

$$\mathbb{T}(v) = \sum_{p \in P} \mathcal{M}(v, p)$$

The *traffic-shift function* ($\mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta)$) represents the fraction of the traffic entering the network at ingress node v that switches egress points after the topology change δ.

$$\mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta) = \frac{1}{\mathbb{T}(v)}. \sum_{p \in P} \mathcal{H}(G, \mathcal{E}(p), v, \delta). \mathcal{M}(v, p)$$

The fraction of traffic that changes egresses may experience transient performance degradation during convergence and changes in forwarding path characteristics (such as congestion, longer RTTs, or packet filters).

We now define data plane sensitivity metrics as a function of the traffic-shift function.

Ingress Node Traffic Sensitivity

The *ingress node traffic sensitivity* ($\eta^{TM} \in [0, 1]$) describes the expected fraction of the traffic originating at ingress node v that switches egress points when considering the set of topology changes in ΔG with probability $\mathbb{P}$. This metric captures the expected variation on v's entry of the traffic matrix for all topology changes in ΔG. We also define η^{TM}_{max} to represent the largest traffic shift experienced by each node v when considering the set of topology changes ΔG.

$$\begin{aligned}
\eta^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}, v) &= \sum_{\delta \in \Delta G} \mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta). \mathbb{P}(\delta) \\
\eta^{TM}_{max}(G, P, \Delta G, \mathcal{M}, v) &= \max_{\delta \in \Delta G} \mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta) \\
\hat{\eta}^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}) &= \frac{1}{|\mathcal{I}|}. \sum_{v \in \mathcal{I}} \eta^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}, v)
\end{aligned}$$

η^{TM} can also be computed by averaging the total volume of traffic shifts across all egress sets and all possible topology changes over the total traffic originated at v. As discussed with routing sensitivity metrics, traffic sensitivity ranges from zero (indicating that traffic from an ingress point does not shift egress points when considering ΔG) to

one (the other extreme, where any change in ΔG causes all the traffic originated at v to switch egress points).

Traffic Impact of a Topology Change

The *traffic impact of a topology change* ($\theta^{TM} \in [0, 1]$) represents the average across all ingress points of the fraction of the traffic that shifts because of a topology change δ. It captures the variation in the traffic matrix after the topology change δ.

$$\begin{aligned}
\theta^{TM}(G, P, \mathcal{M}, \delta) &= \frac{1}{|\mathcal{I}|} \cdot \sum_{v \in \mathcal{I}} \mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta) \\
\theta^{TM}_{max}(G, P, \mathcal{M}, \delta) &= \max_{v \in N} \mathcal{H}^{TM}(G, P, \mathcal{M}, v, \delta) \\
\hat{\theta}^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}) &= \sum_{\delta \in \Delta G} \theta^{TM}(G, P, \mathcal{M}, \delta).\mathbb{P}(\delta)
\end{aligned}$$

These metrics represent the fraction of the volume of traffic that shifts egress points due to an intradomain routing change. θ^{TM}_{max} represents the traffic shift experienced by the node that experiences the largest traffic shift because of a topology change. By computing this metric, network administrators can make statements such as, "no single link failure will shift more than 1% of the traffic".

Overall Data Plane Sensitivity

The *overall data plane sensitivity* ($\sigma^{TM} \in [0, 1]$) describes the average ingress node traffic sensitivity or the expected traffic impact of a topology change. It captures the average change in the traffic matrix.

$$\begin{aligned}
\sigma^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}) &= \hat{\theta}^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M}) \\
&= \hat{\eta}^{TM}(G, P, \Delta G, \mathbb{P}, \mathcal{M})
\end{aligned}$$

The maximum data plane sensitivity (σ^{TM}_{max}) represents the worst case traffic shift experienced by a node considering all topology changes in ΔG.

$$\begin{aligned}\sigma^{TM}_{max}(G, P, \Delta G, \mathcal{M}) &= \max_{v \in N} \eta^{TM}_{max}(G, P, \Delta G, \mathcal{M}, v) \\ &= \max_{\delta \in \Delta G} \theta^{TM}_{max}(G, P, \mathcal{M}, \delta)\end{aligned}$$

Together the average and the maximum data plane sensitivity metrics allow us to compare different networks with respect to their robustness in the flow of traffic under intradomain routing changes. Network administrators may also consider analyzing traffic for specific customers or applications separately. For customers using interactive applications any traffic shift may cause performance degradation. By studying data plane sensitivity for the subset of the addresses corresponding to those customers, either the network can be reprovisioned or the customer connectivity location can be changed to minimize disruptions due to internal events.

5.5 Applying the Model

This section demonstrates the utility of our model and metrics by analyzing the sensitivity of a large AS of a tier-1 ISP network to link and router failures. First, we give a brief explanation of how to obtain the input parameters for the model from measurements collected from operational networks. Then, we analyze the control plane sensitivity of the tier-1 AS. An analysis of the data plane sensitivity of the network remains future work.

5.5.1 Obtaining Inputs for the Model

Most large ISPs routinely collect routing and traffic measurement data for network management purposes. One can leverage this data to extract the input parameters

for our model as follows:

- The **network topology (G)** and the **ranking function (d)** can be derived either from snapshots of a router's IGP configuration or from archives of IGP routing messages collected by a route monitor.

- The **set of destination prefixes (P)** and **prefix-to-egress-set mapping ($\mathcal{E}$)** can be computed by joining a collection of BGP tables or from archives of BGP routing messages.

- Snapshots of the **traffic demands** $\mathcal{M}$ can either be estimated from link load statistics [MTSD02, ZRLD03] or can be measured directly at ingress routers [FGL+01, LPC+04].

The ISP uses OSPF as its intradomain protocol and the network has been partitioned into several areas [Moy98]. We use the OSPF monitor described in Chapter 3 to collect link-state advertisements from the network. The monitor has a direct physical connection to a router in area 0. Thus, it receives detailed information about all links and routers in area 0, but only summarized information about routers in other areas. The summarized information consists of the distance to each router in the non-zero area from the border routers between area 0 and the non-zero area in question. We use the area 0 topology and the summarized information to construct snapshots of the network topology and to extract the ranking function d for each router in area 0. Note that even though we do not have detailed topology information for non-zero areas, the area 0 topology and summarized information allow us to compute the exact OSPF distance from any router in area 0 to any other router in the network.

We use the BGP monitor presented in Chapter 3 to compute the egress set per prefix. For each destination prefix p, we compute $\mathcal{E}(p)$ as the union of all egress points selected by each of the route reflectors monitored. There may be routes learned at some border routers that are not announced internally. In this situation no other router in the

network can use these border routers as egress points, so it will not impact our sensitivity measurements. Since we do not model the route-reflector hierarchy, we use the same mapping from prefix to egress set $\mathcal{E}$ for all routers in the network.

5.5.2 Case Study: An ISP Network

This section analyzes the control sensitivity of an AS in an ISP network. First, we present an in-depth analysis of control plane sensitivity that focuses on a snapshot of the AS collected on June 1, 2004. Then, we evaluate how control plane sensitivity evolves over time.

Control Plane Sensitivity Analysis

We illustrate how a network administrator could use our model to analyze the control plane sensitivity of the network to internal routing changes. In the absence of accurate failure models for the network, network administrators usually consider simple failure scenarios such as link and router failures. We consider two sets of topology changes: $\Delta_L G$ is the set of all single link failures and $\Delta_R G$ is the set of all single router failures (excluding failures of the egress points). We assume that all failures within each set happen with equal probability. Because we only have detailed topology information for area 0 of the AS, we only consider failures in area 0 links and routers. Furthermore, we focus only on area 0 routers to compute control plane sensitivity.

We proceed with an exercise that shows how a hypothetical network administrator could use our model to answer queries of different aspects of the network sensitivity to link and router failures.

How sensitive is the network to single link and single router failures?

Table 5.6 presents the overall control plane sensitivity of the network to single link and single router failures. We compute lower bound of control plane sensi-

tivity (σ^{RM}) by using the best case routing shift function ($\mathcal{H}^{RM}_{best}$) as defined in Section 5.4.1. Similarly, the upper bound is computed using the worst case routing shift function ($\mathcal{H}^{RM}_{worst}$). For conciseness of notation, we denote control plane sensitivity to link failures as $\sigma^{RM}(\Delta_L G)$ and to router failures as $\sigma^{RM}(\Delta_R G)$. Both $\sigma^{RM}(\Delta_L G)$ and $\sigma^{RM}(\Delta_R G)$ are very close to zero, indicating that overall the network is quite robust to single link and router failures. The network is relatively more sensitive to router failures than link failures ($\sigma^{RM}(\Delta_L G) < \sigma^{RM}(\Delta_R G)$). This matches the intuition that the failure of a router should impact more paths than the failure of a single link. Note that σ^{RM} represents control plane sensitivity that is averaged over all routers and all failures; having a low value for both single link and router failures means that the network on average is robust to these set of failures. However, there may be some "outliers" from the average that could still perturb the network. The next question explores this issue further.

	Overall sensitivity (σ^{RM})	
	Lower Bound	*Upper Bound*
Single link failures ($\Delta_L G$)	0.00002	0.00010
Single router failures ($\Delta_R G$)	0.01001	0.01165

Table 5.6: Overall control plane sensitivity of the AS.

What is the largest disruption that can happen in the network?

Table 5.7 presents the worst case control plane sensitivity σ^{RM}_{max}. Recall that σ^{RM}_{max} represents the sensitivity of the pair (v, δ) that has the maximum routing sensitivity in the network. The upper-bound sensitivity to both $\Delta_L G$ and $\Delta_R G$ is almost 1. This indicates that there is at least one router that is extremely sensitive to at least one of the link and router failures. Assume that v_{max}, δ_{max} represents the worst case scenario. Values of $\sigma^{RM}_{max} = 1$ means that v_{max} shifts egresses for *all* destination prefixes as a consequence of the topology change δ_{max}. This leads to the conclusion that, although

on average the network is robust to single link and router failures, there exist a set of routers that are extremely sensitive to certain failures.

	Worst case sensitivity (σ_{max}^{RM})	
	Lower Bound	*Upper Bound*
Single link failures ($\Delta_L G$)	0.5586	0.9974
Single router failures ($\Delta_R G$)	0.9974	0.9974

Table 5.7: Overall control plane sensitivity of the AS.

Our hypothetical administrator digs deeper by slicing the data cube per failure and computing the impact of individual failures, and then slicing it per router and computing each router's routing sensitivity.

Which failures are most disruptive?

Figure 5.7 shows the routing impact (θ^{RM}) of link and router failures. The top of the bar in the plot is the upper bound sensitivity computed using $\mathcal{H}_{worst}^{RM}$ as defined in Section 5.4.1, the bottom is the lower bound computed using $\mathcal{H}_{best}^{RM}$, and the middle point is the average of the two (we use this notation on all the remaining plots in this case study). The x-axis is the fraction of topology changes sorted according to the upper-bound routing impact. The average routing impact of both link and router failures is low. For instance, $\theta^{RM} < 0.01$ for 93% of link failures and 59% of router failures. It is clear from this plot that there are a few failures that have a considerable impact on some routers. In particular, some router failures have $\theta^{RM} > 0.2$, which means that they impact an average of 20% of the destination prefixes across all routers. Although the impact of link failures is lower, there may be some routers that experience large route shifts because of some link failures. After determining the most disruptive failures, the administrator can use this information while making decisions on traffic engineering and network provisioning. The next step is to understand which routers are sensitive to hot-potato disruptions.

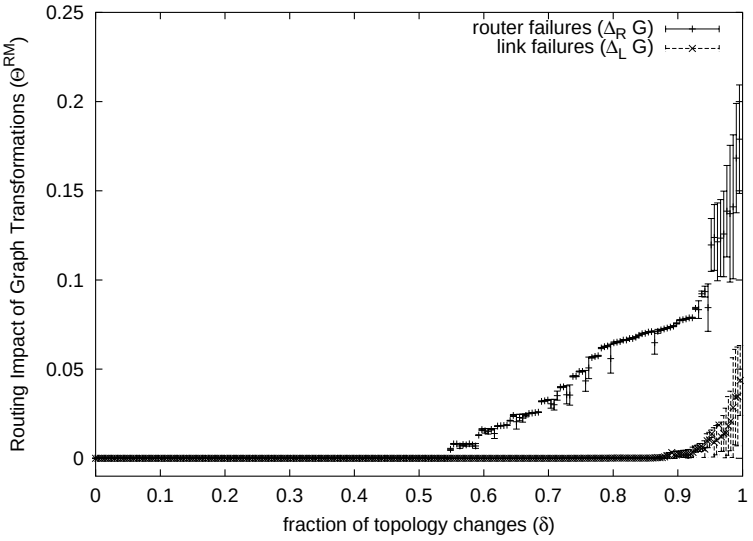


Figure 5.7: Routing impact of router and link failures.

Which routers are most sensitive?

Figure 5.8 shows the upper and lower bounds for node routing sensitivity to link and router failures. The x-axis is the fraction of all routers in area 0 sorted according to their upper-bound routing sensitivity. Almost all the routers experience very few routing changes on average. The average sensitivity may not be the best indicator: a router that experiences small route shifts for a number of failures may have the same node routing sensitivity (η^{RM}) as another router that experiences a very large route shift for only one of the failures. The latter case is arguably more disruptive than the former.

It is also interesting to note that there is a high variance among routers. A router's sensitivity to internal changes depends on its location relative to the closest and second-closest egress points for most destination prefixes. This variance of routing sensitivity across routers is consistent with the empirical findings presented in Chapter 4. Indeed, if we rank routers based on the number of hot-potato changes as measured by the algorithm presented in Chapter 3, and rank the same routers according to their node routing sensitivity to single link failures, we find that the ranking is the same. For single

router failures, however, the ranks do not agree. Although this might sound counter-intuitive, our metrics depend on the failure model we use. Given that router failures are rare events in practice, it is not surprising that the empirical results are more consistent with single link failure sensitivity metrics.

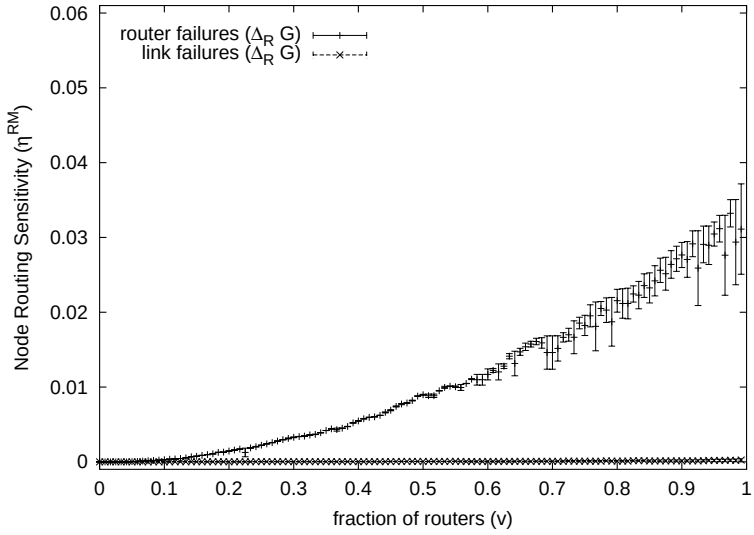


Figure 5.8: Node routing sensitivity to router and link failures.

We observe that node routing sensitivity on average is very low, which is not surprising since we consider all failures to be equally probable and η^{RM} represents the average sensitivity over all possible failures. Next, the administrator investigates node routing sensitivity further by analyzing the worst case routing shift for each router.

What is the largest routing shift experienced by each router?

Figures 5.9 and 5.10 present the worst case node routing sensitivity η^{RM}_{max} for single router failures and for single link failures, respectively. Over 30% of the routers experience considerable hot-potato disruptions for at least one of the router failures.[2] However, the impact of a particular router failure is usually limited to a few routers (usu-

[2]Note that many of these routing changes may be unavoidable without overloading the links leading to the old egress points. We discuss these trade-offs in Section 5.6.

ally located in the same PoP). Fewer routers experience hot-potato disruptions caused by link failures when compared to those caused by router failures.

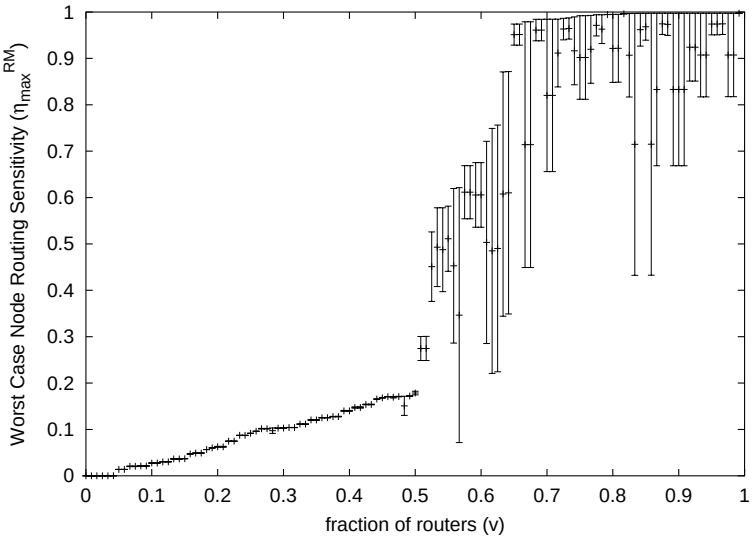


Figure 5.9: Worst case node routing sensitivity to router failures.

It is important to note the wide gap between lower and upper bounds for some nodes in Figures 5.9 and 5.10. This suggests that a substantial fraction of routing shifts for each router depends on the non-deterministic tie breaking step in the BGP decision process.

Note that the order of the routers in Figure 5.9 is different than that presented in Figure 5.10. Comparing node routing sensitivity to router failures with that of link failures for each router v, we find that 49% of the routers are more sensitive to link failures than router failures. Although counter-intuitive at first, such scenarios may arise in practice because hot-potato changes only occur when a topology change (link or router failure) changes the relative distance from the router to the closest and second-closest egress points. Consider the example presented in Figure 5.11, where we assume that A and B are egress points for all destination prefixes. Node E will shift all its routes from egress A to B upon the failure of link AC, whereas the failure of node C does not

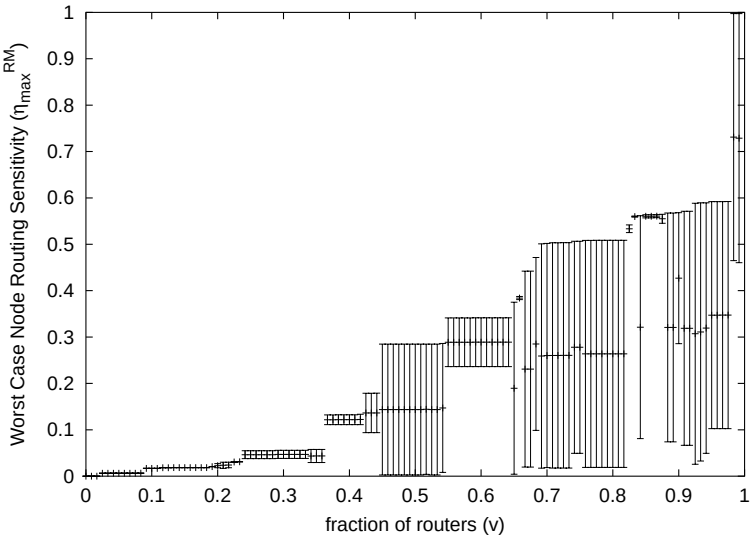


Figure 5.10: Worst case node routing sensitivity to link failures.

cause any egress shift, only the internal path changes to use router D. This indicates that optimizing the network topology or configuration to minimize sensitivity to one type of topology changes may result in an increase in the sensitivity to some other type of change.

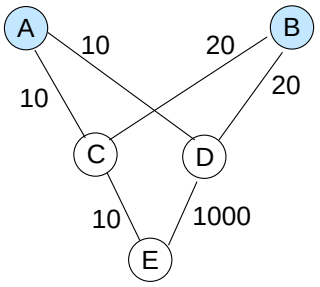


Figure 5.11: Example showing higher sensitivity to a single link failure than a single router failure.

We analyze the most sensitive routers further. We call the router with highest $\eta_{max}^{RM}(\Delta_R G)$ (rightmost router in Figure 5.9) router A and the router with highest $\eta_{max}^{RM}(\Delta_L G)$ router B. Figures 5.12 and 5.13 present the distribution of route shifts

($\mathcal{H}^{RM}$) for routers A and B, respectively. Only a very small fraction of the failures cause the worst case hot-potato disruption. Overall, even the most sensitive routers are robust to most changes in the network; only a small fraction of failures cause large routing shifts.

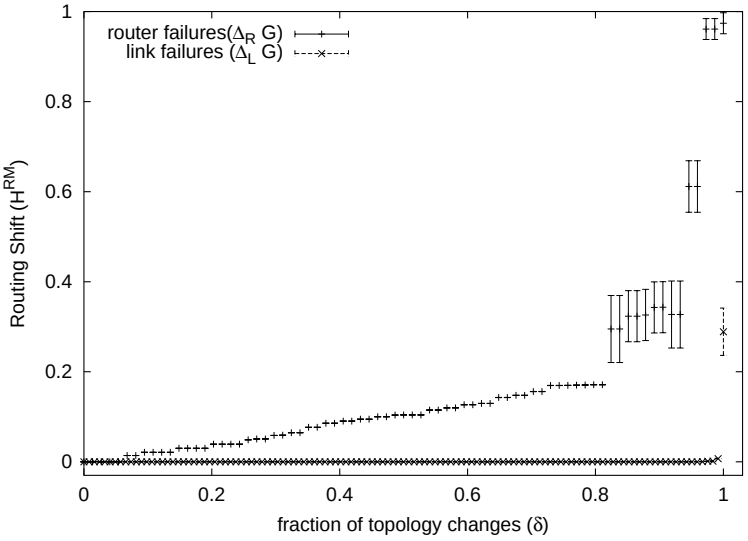


Figure 5.12: Distribution of control plane sensitivity of router A.

Network administrators can use the knowledge of which routers are more sensitive to hot-potato disruptions when deciding in which location to connect customers. For instance, customers that use interactive applications such as VoIP or gaming are more sensitive to the forwarding instabilities caused by a hot-potato change. An ISP may decide that connecting a customer in a less sensitive location may be worth the cost of a long-haul link.

Temporal Variation of Sensitivity

The sensitivity analysis presented in the previous section focused on one snapshot of the network state. A large tier-1 ISP, however, has hundreds of routers and links and consequently the state of the network is in constant flux due to failures and

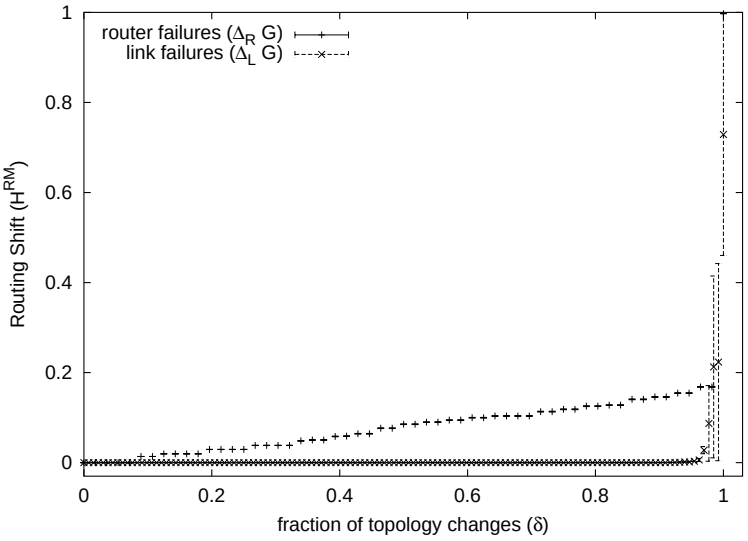


Figure 5.13: Distribution of control plane sensitivity of router B.

maintenance activities. This makes the selection of a representative snapshot of the network particularly challenging. We now study one snapshot of $(G, P, \mathcal{E})$ per month from February 2003 to June 2004 to determine the sensitivity of our analysis to the choice of the network snapshot and the variation of control plane sensitivity over time.

Figure 5.14 presents the overall control plane sensitivity σ^{RM} to both $\Delta_L G$ and $\Delta_R G$ for one snapshot per month during this 17-month period. The network's overall control plane sensitivity to both types of failures is low. As seen earlier, the network is relatively more sensitive to router failures (from 0.012 to 0.021) than link failures (from 0.00003 to 0.0001). We also observe that there is no dramatic variation of σ^{RM} during this 17-month period, just a small decrease between months one and two, another between eight and nine, and a decrease in the last month. This decrease in overall control plane sensitivity indicates that on average the network has become more robust to router and link failures over time.

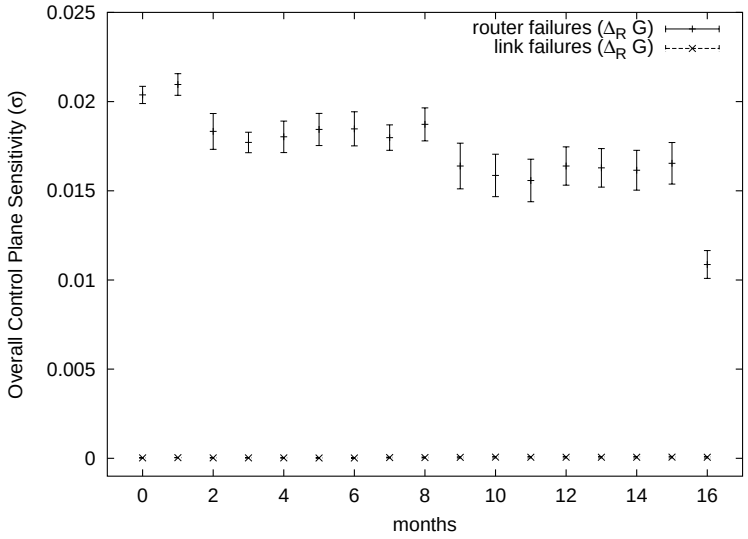


Figure 5.14: Overall sensitivity to router and link failures over time.

5.6 Recommended Operational Practices

Avoiding hot-potato routing changes helps prevent shifts in traffic, extra delays in forwarding-plane convergence, and externally-visible BGP updates. This can improve the end-to-end performance of Internet traffic flowing through the AS. Our analysis of the control plane sensitivity shows that, on average, the ISP network is very robust to link and router failures. Nevertheless, there is room for improvement: some link and router failures can cause routers to shift egress points for a large number of destination prefixes. After identifying the most sensitive routers and the most disruptive failures, network administrators and designers can use this information to improve network robustness. In doing so, there are some design guidelines and operational practices that should be considered to minimize the impact of internal routing changes on egress point selection.

Link and node redundancy

One approach to minimize sensitivity is to replicate all paths in the network. Clearly, this approach is too expensive in practice. However, our analysis shows that

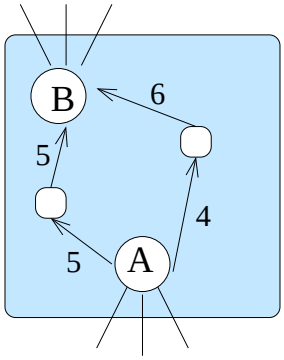


Figure 5.15: Preventing hot-potato routing changes with redundant paths.

there are a few critical links and routers. Replicating these network components can help minimize hot-potato disruptions. As shown in Figure 5.15, router A has two shortest paths (with an IGP distance of 10) to egress point B. This decreases the likelihood that a single internal event would change the IGP distance to reach B and, as such, would tend to avoid hot-potato changes in the BGP routes. Having multiple shortest paths between pairs of routers is also useful to reduce the latency for forwarding-plane convergence for IGP routing changes [SMD03], even when no BGP-level change occurs. Our model can be used iteratively to determine the network sensitivity after the addition of new components.

Selection of peering and service locations

Another approach to minimize overall sensitivity is to have only one egress point or to have peering at every router. Neither of these solutions is desirable for practical reasons: (i) peering locations depend on business relations and it is not feasible for an ISP to peer with all other ISPs in every location; and (ii) selecting only one peering location for each destination prefix is not desirable from a reliability and traffic engineering perspective. ISPs can use the knowledge of which locations of the network are more sensitive to disruptions and prioritize adding connections to peers at these locations.

ISPs can also avoid connecting more sensitive services such as VoIP or gam-

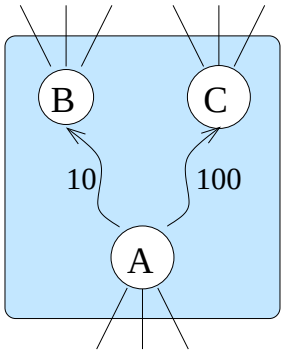


Figure 5.16: Preventing hot-potato routing changes by having a preferred egress router.

ing to locations that have higher values of node routing control plane sensitivity, as discussed in Section 5.5.2. For example, imagine that our sensitivity analysis shows that the San Diego PoP is more likely to experience hot-potato disruptions. As a result, the administrator may decide to connect a gaming server in Los Angeles instead.

Reconfiguring the network

If each router had a single nearest egress point for reaching most destination prefixes, then the impact of hot-potato disruptions would be reduced. As shown in Figure 5.16, router A has a small IGP distance of 10 to reach egress point B and a much larger IGP distance of 100 to reach C. This reduces the likelihood that small variations in IGP distances would trigger a hot-potato routing change at A. Only a very dramatic internal network event would cause A to choose egress point C over egress point B. However, selecting the best configuration of link weights to reduce network sensitivity to all routers in the network adds an extra dimension to the problem of optimizing link weights for traffic engineering, which is already know to be NP-complete [FRT02]. Moreover, optimizing link weights for one type of topology changes may increase sensitivity for other types. Although more accurate failure models can help guide this analysis, finding an optimal configuration of link weights remains a hard problem.

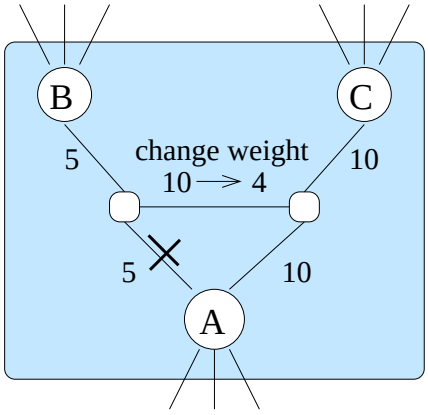


Figure 5.17: A still picks egress B during maintenance

Reconfiguration of link weights can be used to avoiding hot-potato disruptions during planned maintenance activities. In this case, administrators know in advance which topology change will be applied to the graph and when. For example, in Figure 5.17 the router A selects egress point B with an IGP distance of 10 over egress C with a distance 20. However, if the left link from A needs to be disabled for upgrading, the distance to B would increase to 25, making C the closer egress point. The hot-potato routing change can be avoided by changing the weight of the middle link from 10 to 4 before the maintenance activity; this ensures that the path to B has distance 19—smaller than the distance to C. This simple example illustrates how the reconfiguration of link weights can avoid hot-potato changes. Administrators can use our model to assist the identification of a less disruptive configuration of the link weights to be deployed before the event.

It should be clear from this discussion that link and node redundancy, selection of peering and service locatiosn, and the configuration of the network together represent a trade-off that network designers and administrators need to consider when managing their networks. Our sensitivity analysis can assist in making these trade-offs.

5.7 Summary

In this chapter, we develop methods for characterizing network sensitivity to intradomain routing changes, or *hot-potato disruptions*, to ultimately improve network robustness. First, we propose and describe an analytic model of the interaction between IGP and BGP and its impact on both the control and data planes of an ISP network. Based on this model, we define a set of metrics for describing a network's sensitivity to intradomain routing perturbations. We study control plane sensitivity of a large AS of a tier-1 ISP to link and router failures. This analysis demonstrates the utility of our model for identifying which routers are particularly sensitive to internal perturbations and which failures would cause most disruptions.

Our approach for building more robust networks has focused on improving network design given routing protocols as they exist today. Network robustness problems are intrinsic in the way BGP reacts to small IGP distance changes. We now propose a mechanism to replace hot-potato routing that allows network administrators to select more robust egress selection policies.

Chapter 6

Tunable Interdomain Egress Selection

This chapter proposes a new egress selection mechanism to replace hot-potato routing that allows a wide range of egress selection policies. Currently, the policy of selecting the closest egress point is hard-coded in the BGP decision process implemented on each router [RLH04]. Hot-potato routing is an appealing mechanism for two main reasons. First, hot-potato routing tends to limit the consumption of bandwidth resources in the network by shuttling traffic to the next AS as early as possible. Second, under hot-potato routing, a router's choice of egress point is guaranteed to be consistent with the other routers along the forwarding path, because packets are forwarded to neighboring routers that have selected a BGP route with the same (closest) egress point.

Although consistent forwarding is clearly an important property for any routing system, routers now have other ways of achieving this goal. In particular, the greater availability of tunneling technology allows for more sophisticated egress-selection rules, which are not tied to the IGP metrics. Internet Service Providers (ISPs) increasing use tunneling technologies—such as IP-in-IP encapsulation or MultiProtocol Label Switching (MPLS)—to support Virtual Private Networks (VPNs) or to avoid running BGP on their internal routers. We capitalize on tunneling techniques to revisit the hard-coded policy of selecting egress points based on IGP distances, because we believe that hot-potato routing is:

- **Too restrictive:** The underlying mechanism dictates a particular policy rather than supporting the diverse performance objectives important to network administrators.
- **Too disruptive:** Chapter 4 shows that small changes in IGP distances can sometimes lead to large shifts in traffic, long convergence delays, and BGP updates to neighboring domains.
- **Too convoluted:** Network administrators are forced to select IGP metrics that make "BGP sense," rather than viewing the two parts of the routing system separately.

Selecting the egress point and computing the forwarding path to the egress point are two very distinct functions, and we believe that they should be decoupled. Paths inside the network should be selected based on some meaningful performance objective, whereas the egress selection should be flexible to support a broader set of traffic-engineering goals. These objectives vary by network and destination prefix; therefore a mechanism that imposes a single egress selection policy cannot satisfy these diverse set of requirements. We believe that the decision to select egress points based on IGP distances should be revisited, in light of the growing pressure to provide good, predictable communication performance for applications such as voice-over-IP, online gaming, and business transactions.

In this chapter, we propose a new way for each router to select an egress point for a destination by comparing the candidate egress points based on a weighted sum of the IGP distance and a constant term. The configurable weights provide flexibility in deciding whether (and how much) to base BGP decisions on the IGP metrics. Network management systems can apply optimization techniques to automatically set these weights to satisfy network-level objectives, such as balancing load and minimizing propagation delays. To ensure consistent forwarding through the network, our mechanism

requires the use of tunnels to direct traffic from the ingress router to the chosen egress point. Our new mechanism, called TIE (Tunable Interdomain Egress) because it controls how routers break ties between multiple equally-good BGP routes, is both simple (for the routers) and expressive (for the network administrators). Our solution does not introduce any new protocols or any changes to today's routing protocols, making it possible to deploy our ideas at one AS at a time and with only minimal changes to the BGP decision logic on IP routers.

In the next section, we summarize the problems caused by hot-potato routing, and describe an alternative where each router has a fixed ranking of the egress points. Then, Section 6.2 presents the TIE mechanism for selecting egress points, along with several simple examples. Section 6.3 presents the two networks that are studied in this chapter. Sections 6.4 and 6.5 present the two optimization problems and evaluate our solutions on topology, traffic, and routing data from two backbone networks. In Section 6.6, we discuss how to limit the number of configurable parameters and how to deploy TIE without changing the existing routing protocols. We summarize the chapter in Section 6.7.

6.1 Critique of Today's IGP/BGP Boundary

The Internet routing system has three main components: (i) interdomain routing, which determines the set of border (or *egress*) routers that direct traffic toward a destination, (ii) intradomain routing, which determines the path from an ingress router to an egress router, and (iii) egress-point selection, which determines which egress router is chosen by each ingress router for each destination. In this section, we first describe how tying egress selection to IGP distances leads to harmful disruptions and over-constrained traffic-engineering problems. Then, we explain how the alternative of allowing each ingress router to have a fixed ranking of egress points is not flexible enough (for traffic engineering) or adaptive enough (to large changes in the network topology).

Our discussion of the two approaches draws on the example network in Figure 6.1. AS 1 has five routers (A, B, C, D, and E) and each internal link has an IGP metric. Router C learns BGP routes to destination p from egress routers A and B. Under hot-potato routing, router C chooses the BGP route learned from A because the IGP distance to A is 2, which is smaller than the distance of 9 to B. However, if the CD link fails, all traffic from C to p would shift to egress router B, with an IGP distance of 9 that is smaller than the IGP distance of 10 to A. In this section, we argue that these kinds of routing changes are disruptive. Yet, continuing to use egress-point A might not be the right thing to do, either, depending on the propagation delay, traffic demands, and link capacities. Instead, network administrators need a mechanism that is flexible enough to support sound performance trade-offs. Neither hot-potato routing nor fixed ranking allow these trade-offs to be made.

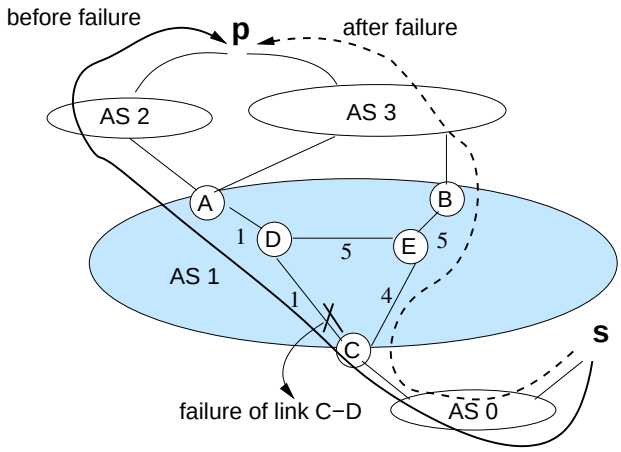


Figure 6.1: Example of an internetwork.

6.1.1 Hot-Potato Routing

Hot-potato routing adapts automatically to topology changes that affect the relative distances to the egress points. Although hot-potato routing seems like a reasonable way to minimize resource consumption, IGP link weights do not express resource

usage directly. The IGP distances do not necessarily have any relationship to hop count, propagation delay, or link capacity, and selecting the closer egress point does not necessarily improve network performance. In addition, small topology changes can lead to performance disruptions:

- **Large shifts in traffic within and between ASes:** We showed in Chapter 4 that a single link failure can affect the egress-point selection for tens of thousands of destinations at the same time, leading to large shifts in traffic. In fact, hot-potato routing changes are responsible for many of the largest traffic variations in a large backbone.

- **Changes in the downstream path:** When the egress point changes, the traffic moves to a different downstream forwarding path that may have a different round-trip time or available bandwidth, which may disrupt the communicating applications. In addition, the abrupt increase in traffic entering the neighboring AS may cause congestion.

- **BGP update messages for neighboring domains:** A change in egress point may also change the AS path. The failure of the CD link in Figure 6.1 causes router C to switch from a path through AS 2 to one through AS 3, forcing C to send a BGP update message to AS 0. Global BGP convergence may take several minutes [LABJ01]. If AS 0 switches to a BGP route announced by another provider, the traffic entering AS 1 at router C would change.

Even if the hot-potato routing change does not lead to new BGP update messages, long convergence delays can occur inside the AS depending on how the router implements the BGP decision process. Section 4.1.1 presented long convergence delays because the underlying routers in the network only revisited the influence of IGP distances on BGP decisions once per minute; during the convergence period, data packets may be lost, delayed, or delivered out of order. This particular problem, while serious, can be ad-

dressed by having routers use an event-driven implementation that immediately revisits the BGP routing decisions after a change in the intradomain topology. In contrast, the three problems listed above are fundamental.

In a large network, IGP changes that affect multiple destination prefixes happen several times a day, sometimes leading to very large shifts in traffic. Not all of these events are caused by unexpected equipment failures—a large fraction of them are caused by planned events, such as routine maintenance. Maintenance activities happen very frequently to upgrade the operating system on the routers, replace line cards, or repair optical amplifiers. In addition, construction activities may require moving fibers or temporarily disabling certain links. A recent study of the Sprint backbone showed that almost half of IGP events happened during the maintenance window [ICBD04]. Often, shifts in egress points are not necessary. The new intradomain path to the old egress point, although a little longer IGP-wise, may offer comparable (or even better) performance than the path to the new egress point. Following the failure of the CD link in Figure 6.1, the path C, E, D, A might be less congested or have lower propagation delay than the path C, E, B. Moreover, many internal network changes are short-lived; a study of the Sprint backbone showed that 96% of failures were repaired in less than 15 minutes [ICBD04]. Maintenance activities are often done in periods of lower traffic demands, when the network would comfortably have extra capacity to tolerate the temporary use of non-closest egress points.

Besides being disruptive, the tight coupling between egress selection and IGP metrics makes traffic engineering and maintenance planning extremely difficult. Network administrators *indirectly* control the flow of traffic by tuning the IGP metrics [Rex05, FT02, LORS01, FT00] and BGP policies [FWR04, Uhl04]. Finding good settings that result in the desired behavior is computationally challenging, due to the large search space and the need to model the effects on egress-point selection. Finding settings that are robust to a range of possible equipment failures is even more dif-

ficult [NSTD03, FT03, SG05]. Imposing even more constraints, such as minimizing hot-potato disruptions across all routers and destination prefixes, makes the problem increasingly untenable. In addition, once the local-search techniques identify a better setting of the IGP metrics or BGP policies, changing these parameters in the routers requires the network to go through routing-protocol convergence, leading to transient performance disruptions.

6.1.2 Fixed Ranking of Egress Points at Each Ingress Router

A natural alternative would be to configure each router with a fixed ranking of the egress points, where the router would select the highest-ranked element in the set of egress routers for each destination. This solution can be realized using today's technology by establishing a tunnel from each ingress router to each egress router, and assigning an IGP metric to the tunnel. For example, network administrators can use MPLS [RVC01, DR00] to create label-switched paths (LSPs) between all ingress-egress pairs. Configuring each LSP as an IGP *virtual link* ensures that each tunnel appears in the intradomain routing protocol. The metric assigned to the tunnel would then drive the hot-potato routing decision hard-coded in the routers. The data packets would follow the shortest underlying IGP path from the ingress router to the chosen egress router. The hot-potato mechanism would still dictate the selection of egress points, but the metric associated with each tunnel would be defined statically at configuration time rather than automatically computed by the IGP. Thus, this technique allows network administrators to rank the egress points from each router's perspective, allowing each ingress router to select the highest-ranked egress point independent of internal network events, short of the extreme case where the egress point becomes unreachable and the router is forced to switch to the egress point with the next highest rank.

For the example in Figure 6.1, router C could be configured to prefer egress A over B. Then, when the CD link fails, C would continue to direct traffic toward router

A, though now using the path C, E, D, A. This configuration would avoid triggering the traffic shift to B, changes in the downstream forwarding path, and BGP updates to neighboring domains. However, although the fixed ranking is extremely robust to internal changes, sometimes switching to a different egress point is a good idea. For example, the path C, E, D, A may have limited bandwidth or a long propagation delay, making it more attractive to switch to egress-point B, even at the expense of causing a transient disruption. In the long term, network administrators could conceivably change the configuration of the ranking to force the traffic to move to a new egress point, but the reaction would not be immediate. Similarly, the administrators could reconfigure the IGP metrics or BGP policies to redistribute the traffic load, at the expense of searching for a suitable solution, reconfiguring the routers, and waiting for the routing protocol to converge.

The mechanisms available today for selecting egress points represent two extremes in the trade-off between robustness and automatic adaptation. Hot-potato routing adapts immediately to internal routing changes (however small), leading to frequent disruptions. Imposing a fixed ranking of egress points, while robust to topology changes, cannot adapt in real time to critical events. Neither mechanism offers sufficient control for network administrators trying to engineer the flow of traffic and to plan for maintenance. In this thesis, we ask a natural question: *Is there a mechanism for egress-point selection that is flexible enough to control the flow of traffic in steady state, while responding automatically to network events that would degrade performance?*

6.2 TIE: Tunable Interdomain Egress Selection

In this section, we propose a mechanism for selecting an egress point for each ingress router and destination prefix in a network. Ideally, an optimization routine could compute the egress points directly based on the current topology, egress sets, and traffic, subject to a network-wide performance objective. However, the routers must adapt in

real time to events such as changes in the underlying topology and egress sets, leading us to design a simple mechanism that allows a separation of timescales—enabling both rapid adaptation to unforeseen events and longer-term optimization of network-wide objectives. In addition, the design of our mechanism places an emphasis on generality to allow us to support a wide variety of network objectives, rather than tailoring our solution to one particular scenario. In this section, we first describe our simple mechanism. Then, we present several examples of how to set the configurable parameters to manage a simple network. We end with a high-level description of how network administrators will use TIE.

6.2.1 TIE Ranking Metric

Our mechanism allows each router to have a ranking of the egress points for each destination prefix. That is, router i has a metric $m(i, p, e)$, across all prefixes p and egress points e. For each prefix, the router considers the set of possible egress points and selects the one with the smallest rank, and then forwards packets over the shortest path through the network to that egress point. Our approach differs from the scheme in Section 6.1.2 in several key ways. First, our ranking metric has *finer granularity*, in that we allow an ingress router to have a different ranking for different destination prefixes. Second, our ranking metric is computed rather than statically configured, allowing the ranking to *adapt to changes* in the network topology and egress set. Third, our metric is *not tied directly to the underlying tunnel* that directs traffic from an ingress point to the chosen egress point, allowing us to achieve the finer granularity of control without increasing the number of tunnels. Our approach is also more flexible than tuning BGP routing policies, in that one router can start using a new egress point while other routers continue to use the old one.

To support flexible policy while adapting automatically to network changes, the metric $m(i, p, e)$ must include both configurable parameters and values computed

Undirected graph	$G = (N, L)$, nodes N and links L
Ingress and egress nodes	$i \in N$ and $e \in N$
IGP distance on graph	$d(G, i, e)$, $i, e \in N$
Destination prefix	$p \in P$
Egress set	$\mathcal{E}(p) \subseteq N$
Ranking metric	$m(i, p, e)$, $i, e \in N$, $p \in P$
Tunable parameters	$\alpha(i, p, e)$ and $\beta(i, p, e)$

Table 6.1: Summary of notation.

directly from a real-time view of the topology. We represent intradomain routing topology as an undirected weighted graph $G = (N, L)$, where N is the set of nodes and L is the set of IP links, as summarized in Table 6.1. Based on the link weights, each router $i \in N$ can compute the IGP distance $d(G, i, e)$ to every other router $e \in N$. The egress set $\mathcal{E}(p) \subseteq N$ consists of the edge nodes that have equally-good BGP routes for prefix p. For prefix p, node i selects the egress point $argmin_e\{m(i, p, e) \mid e \in \mathcal{E}(p)\}$. The metric is computed as a weighted sum of the IGP distance and a constant term:

$$m(i, p, e) = \alpha(i, p, e) \cdot d(G, i, e) + \beta(i, p, e),$$

where α and β are configurable values. The first component of the equation supports automatic adaptation to topology changes, whereas the second represents a static ranking of routes for that prefix. Together, these two parameters can balance the trade-off between adaptability and robustness. This simple metric satisfies our three main goals:

- **Flexible policies:** By tuning the values of α and β, network administrators can cover the entire spectrum of egress-selection policies from hot-potato routing to static rankings of egress points. Hot-potato routing can be implemented by setting $\alpha = 1$ and $\beta = 0$ for all nodes and prefixes. A static ranking can be represented by setting $\alpha = 0$ and, for each node i, $\beta(i, p, e)$ to a constant value for all values of p. Our mechanism can also realize a diverse set of policies in between.

- **Simple computation:** The metric is computationally simple—one multiplication and one addition—based on information readily available to the routers (i.e., the IGP distances and the α and β values). This simplicity allows routers to compute the appropriate egress point for all destination prefixes immediately after a change in the network topology or egress set.

- **Ease of optimization:** The mechanism offers two knobs (α and β) that can be easily optimized by a management system based on diverse network objectives. In Sections 6.4 and 6.5, we explore the power of this mechanism to express a wide range of policies, and we demonstrate that it is easy to optimize by showing that the optimization problems we define are tractable.

In addition, when the network-management system changes the α and β values, the affected routers can move traffic from one path to another without incurring any convergence delays. This immediate shift is possible because the network already has forwarding paths between each pair of routers. Changing the α and β values merely changes which paths carry the traffic.

6.2.2 Example Configurations

For each router i and prefix p, network administrators need to configure the values of α and β. By configuring the egress selection parameters on a per prefix basis, an AS can satisfy diverse policy goals. We now explore a few examples:

Voice-over-IP: For instance, suppose that a prefix p is used for VoIP and that network administrators set IGP link weights according to propagation delay. Voice applications are sensitive to both high delays and the transient disruptions that occur during egress-point changes. Imagine that the network learns p at two egress points e_1 and e_2, and that the IGP distance at design time from a router i to each egress is $d(G, i, e_1) = 20$ and $d(G, i, e_2) = 30$. In the designed topology, i should prefer e_1 to forward packets to p to minimize delay. If the cost to reach e_1 increases a little, i should still use e_1 in order

to avoid disruptions associated with the egress change. However, when the IGP distance to e_1 exceeds 50, the network administrators want i to select the closest egress.

This application needs an egress-selection policy that is between hot-potato routing and a fixed ranking. At design time, the value of $m(i,p,e_1) = 20 \cdot \alpha(i,p,e_1) + \beta(i,p,e_1)$ and $m(i,p,e_2) = 30 \cdot \alpha(i,p,e_2) + \beta(i,p,e_2)$. Since i prefers e_1, we need to have $m(i,p,e_1) < m(i,p,e_2)$; however, when $d(G,i,e_1)$ exceeds 50, we need to have $m(i,p,e_1) < m(i,p,e_2)$. We can express these constraints with the following equations:

$$20 \cdot \alpha(i,p,e_1) + \beta(i,p,e_1) < 30 \cdot \alpha(i,p,e_2) + \beta(i,p,e_2)$$

$$50 \cdot \alpha(i,p,e_1) + \beta(i,p,e_1) < 30 \cdot \alpha(i,p,e_2) + \beta(i,p,e_2)$$

$$51 \cdot \alpha(i,p,e_1) + \beta(i,p,e_1) > 30 \cdot \alpha(i,p,e_2) + \beta(i,p,e_2)$$

We can now select the values of α and β that satisfy these constraints. For instance, if we set both $\beta(i,p,e_1) = \beta(i,p,e_2) = 0$ and $\alpha(i,p,e_1) = 10$, then we find that $\alpha(i,p,e_2) = 17$.

Large file transfer: Take now the example of two research labs that continuously exchange large data files. Suppose that each research lab has an ISP and that the two providers peer in two locations. Both the source and the destination ISPs need to provision enough bandwidth for these large transfers. To provision for the file transfers, both ISPs need to know both the ingress and egress points for the data. In this case, the egress selection needs to be stable. Say that the source and destination ISPs agree that e_1 should be responsible for carrying this traffic. Then, for each router i we set $\alpha(i,p,e_1) = \alpha(i,p,e_2) = 0$ and $\beta(i,p,e_1) = 1$ and $\beta(i,p,e_2) = 2$.

Traffic engineering: The first two examples consider a prefix in isolation. However, egress-point selection should also consider network-wide goals. Consider the egress-selection decision for prefixes p_1 and p_2 at router C in Figure 6.2, p_1 is a VoIP prefix and p_2 corresponds to Web servers. In this example, router C has to choose between egresses A and B. Assume that the path with IGP distance 9 has high capacity, whereas the paths with cost 10 and 11 have lower capacity. When all three paths are

	α		β	
	A	B	A	B
p_1	0	0	2	1
p_2	1	1	0	0

Table 6.2: Configuration of parameters for example in Figure 6.2.

working, the network administrators want C to use egress-point B for both prefixes. However, if the path with cost 9 fails, they would like to balance the load over the two lower-bandwidth links. Since the voice traffic to p_1 is sensitive to the routing change, the network administrators would prefer to use B for p_1 and A for p_2. This policy can be implemented by setting the parameters as presented in Table 6.2. C's egress selection to p_1 behaves like a fixed ranking of the egress points, whereas p_2 behaves like hot-potato routing.

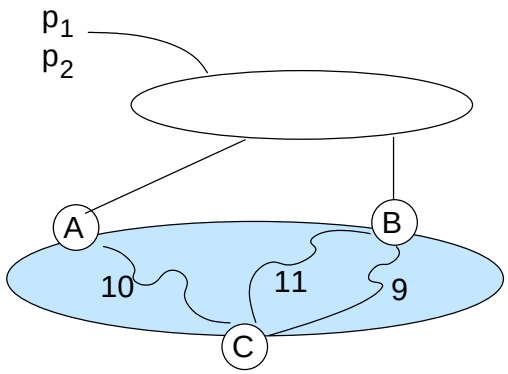


Figure 6.2: Example illustrating heterogeneous traffic types.

Despite the simplicity of this policy, current egress-selection mechanisms cannot express it. Hot-potato routing would cause both p_1 and p_2 to shift to egress A after the path with cost 9 fails, and ranking egress B over A for all prefixes would force all traffic over the low-capacity path with cost 11. Of course, after the failure, the network administrators could change the BGP import policy to p_2 at A to make it look better than B. However, there is a long delay before they can detect the failure and identify the BGP

policy that should be applied to alleviate the problem. Our mechanism allow this policy to be implemented at design time and the network can then adjust itself accordingly.

The setting of α and β can be done independently for each pair (i, p), which leads to easier optimization problems. In contrast, tweaking IGP weights impacts the IGP distance between multiple pairs of routers for all routes, and tweaking BGP policies impacts the preference of all routers in the network for a particular route. One drawback of our mechanism is the large number of parameters that need to be set at each router. We discuss how to configure TIE for large networks next.

6.2.3 Using TIE

We do not envision that network administrators will configure all values of α and β by hand. Instead, we propose an architecture as presented in Figure 6.3. The upper box represents the tasks of a management system that configures the routers, and the lower box captures the tasks running on each router in the network. Network administrators define the high-level goal of the egress-selection policy for the network or for a set of destination prefixes (such as minimizing sensitivity to failures, minimizing delay, or balancing link load). The management system takes as input the current network design and the administrator's specifications, runs an optimization routine to find the appropriate values for the parameters α and β, and configures the routers accordingly. Once the management system configures the TIE parameters, the routers apply the BGP decision process as usual, except for using the metric m to select between multiple equally-good BGP routes.

With TIE the egress-point selection can change for two reasons: high-level policy changes (expressed by changes in α and β) or routing changes. Policy changes happen because of changes in network objectives or the network design. Routing changes—changes in the IGP distances or egress sets—happen in response to network events such as link failures or BGP updates from neighboring domains. Reaction to

routing changes must be done in real time to avoid bad network performance, whereas policy changes happen less often and can be implemented slowly. Our architecture benefits from this separation of timescales. Policy changes require running an optimization routine, which is executed completely off line by the management system running on a separate machine. Under routing or policy changes, routers only need to perform one addition and one multiplication to recompute m. This simple on-line computation also happens under BGP updates. Routers can be pre-configured with default values of α and β for newly announced prefixes. The management system will revisit these values at the time of the next optimization.

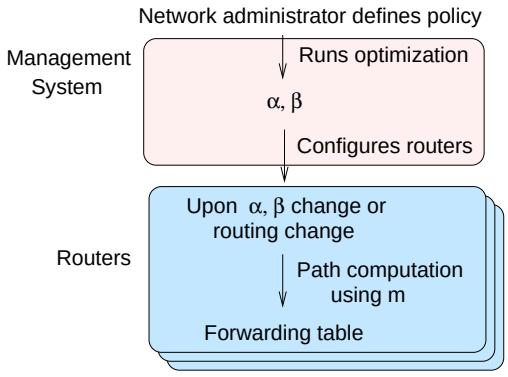


Figure 6.3: A management system optimizes α and β for a high-level policy and configure routers. Routing adapts the egress-point selection at real time in reaction to network events.

In Sections 6.4 and 6.5, we give examples of two useful policies for network administrators. For each of them, we present a management system that selects suitable values of α and β. Then, Section 6.6 addresses implementation issues for deploying TIE. In particular, we discuss techniques for reducing the number of parameters configured in practice and the use of tunnels to allow each router to make independent egress-point selection.

6.3 Configuring TIE for Operational Networks

In Sections 6.4 and 6.5, we evaluate TIE on data from two operational networks. In this section, we present our methodology for obtaining the input data—the internal topology, the egress sets, and the traffic demands—from passive measurements. Since routers in the same PoP essentially act as one larger node, we model the topology of both networks at the PoP level.

6.3.1 Abilene Network

Abilene is the backbone of the U.S. research network [Abi]. The network has 11 PoPs with one router each. The vast majority of the links are OC192, with only one OC48. For our study, we used data from April 2003. We obtained the topology G (both with designed weights and geographic distance) and link capacities $c(l)$ from the publicly-available map of the network. This map has the location of each router, as well as the link capacities and IGP weights.

Each BGP speaker has around $7,500$ prefixes in its routing table. We obtained the egress set $\mathcal{E}(p)$ for each prefix from a dump of the BGP table for a monitor that peers with every router. The network had only 23 distinct egress sets.

We extracted the traffic demands from sampled Netflow data. Every router in the network has Netflow enabled with a sampling rate of $1/100$. For each router i and destination prefix p we have set the traffic demand $\mathcal{M}(i, p)$ to the average traffic volume for one hour of Netflow data collected on a weekday afternoon.

6.3.2 Tier-1 ISP Network

We also used data collected from a tier-1 service-provider backbone on January 10, 2005. We extracted the router-level topology and IGP link weights from the link-state advertisements logged by a routing monitor. We used router configuration data to map each router to a PoP and determine the link capacities. The resulting topol-

ogy has a few dozen nodes. For simplicity, we combine parallel links between a pair of PoPs into one link with the aggregate capacity. We used the PoP locations to determine the geographic distance traversed by each inter-PoP link.

The network learns BGP routes for approximately $150,000$ prefixes. We build the egress set $\mathcal{E}(p)$ for each prefix from the BGP table dumps from all top-level route reflectors in the network. The network has a few hundred distinct egress sets.

We use sampled Netflow data collected around the entire periphery of the network. We aggregate all traffic entering at the same PoP i and destined to the same prefix p into a single traffic demand $\mathcal{M}(i, p)$. Each traffic demand represents the average traffic rate over the course of the day.

6.4 Minimizing Sensitivity to Equipment Failures with Bounded Delay

In this section, we present a prototype of a management system to select values of α and β to minimize the sensitivity of egress-point selection to equipment failures, subject to restrictions on increasing the propagation delay. After presenting a precise formulation of the problem, we present a solution that has two phases—simulating the effects of equipment failures to determine the constraints on the α and β values, and applying integer-programming techniques to identify optimal settings. Then, we evaluate the resulting solution using traffic, topology, and routing data from the two backbone networks described in the previous section.

6.4.1 Problem Definition: Minimizing Sensitivity

Consider a well-provisioned backbone network that supports interactive applications, such as voice-over-IP and online gaming. The network administrators want to avoid the transient disruptions that would arise when an internal failure causes a change

in the egress point for reaching a destination, as long as continuing to use the old egress point would not incur large delays. By setting the IGP link weights according to geographic distance, the shortest IGP path between two nodes would correspond to the smallest delay and the closest egress point would be the best choice. Hence, for this problem, the best egress point $b(G, i, p)$ for node i and prefix p is the node $e \in \mathcal{E}(p)$ with the smallest IGP distance $d(G, i, e)$. If an internal failure occurs, the administrators want node i to continue directing traffic to $b(G, i, p)$ unless the delay to this egress point exceeds $T \cdot d(G, i, b(G, i, p))$ for some threshold $T > 1$. If the delay to reach the egress point exceeds the threshold, the administrators want node i to switch to using the (new) closest egress point to minimize the propagation delay. Table 6.3 summarizes the notation.

Threshold for tolerable delay	T
Set of topology changes	ΔG
Topology change	$\delta \in \Delta G$
Network topology after change	$\delta(G)$
Best egress point for (i, p) on G	$b(G, i, p)$

Table 6.3: Notation for the problem of minimizing sensitivity to topology changes with bounded delay.

In an ideal world, the routers could be programmed to implement this policy directly. For example, upon each IGP topology change δ, each node i could revisit its egress selection for each prefix by performing a simple test for the new topology $\delta(G)$:

if $(d(\delta(G), i, b(G, i, p)) \leq T \cdot d(G, i, b(G, i, p)))$,
  then $b(\delta(G), i, p) = b(G, i, p)$
  else $b(\delta(G), i, p) = argmin_e\{d(\delta(G), i, e) \mid e \in \mathcal{E}(p)\}$.

Modifying every router in the network to implement this egress-selection policy would

guarantee that the network always behaves according to the specified goal. However, adding too many decision rules directly in the routers would be extremely complicated, and ultimately network administrators would want to apply a policy that is not supported in the routers. In the next subsection, we show that TIE is expressive enough to implement this policy. Instead of having the routers apply the test in real time, the network-management system configures the TIE parameters at design time based on the policy, and the routers adapt automatically when internal changes occur.

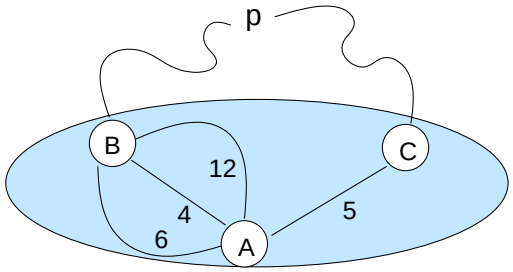
6.4.2 Solving the Sensitivity Problem with TIE

Solving the problem with our mechanism requires us to find values of $\alpha(i, p, e)$ and $\beta(i, p, e)$, for each $i, e \in N$ and $p \in P$, that lead to the desired egress-point selections over all topology changes ΔG. Our solution has two main steps. First, a *simulation phase* determines the desired egress selection both at design time (under graph G) and after each topology change (under graph $\delta(G)$). The output of this phase is a set of constraints on the α and β values for each (i, p) pair. Then, an *optimization phase* determines the values of α and β that satisfy these constraints. For this problem, the egress-point selection for each (i, p) pair is independent.

Simulation Phase

To illustrate how we construct the constraints on α and β for the initial topology G and each topology change δ, consider the example in Figure 6.4(a). In the initial topology, node A would select node B as the egress point because B is closer than C. We can express this by $m(A, p, B) < m(A, p, C)$ for topology G, as shown by the first constraint in Figure 6.4(b). Then, we consider each topology change δ and determine the preferred egress selection with the policy in mind, where $T = 2$ and δ_1 is the failure of the link with cost 4 and δ_2 is the failure of the links with costs 4 and 6. In the new graph $\delta_1(G)$, A is closer to C (with a distance $d(\delta_1(G), A, C)$ of 5) than to B (with a dis-

tance $d(\delta_1(G), A, B)$ of 6). However, since $d(\delta_1(G), A, B) < 2 \cdot d(G, A, B)$, A should continue to select egress-point B. The second equation in Figure 6.4(b) expresses this decision. We use the same methodology to evaluate the best egress selection after δ_2. In this case, the distance from A to B is above the threshold, so A should switch to using egress-point C, as expressed by the third equation.



(a)

Constraints for (A,p):

$4 \cdot \alpha_B + \beta_B < 5 \cdot \alpha_C + \beta_C$

$6 \cdot \alpha_B + \beta_B < 5 \cdot \alpha_C + \beta_C$

$12 \cdot \alpha_B + \beta_B > 5 \cdot \alpha_C + \beta_C$

(b)

Figure 6.4: Example illustrating constraints on values of α and β.

More generally, our algorithm consists of two main steps. First, we compute the distances $d(\cdot, i, e)$ for the original graph G and all changes $\delta \in \Delta G$ using an all-pairs shortest path algorithm. (For simple topology changes, such as all single-link failures, an incremental Dijkstra algorithm can reduce the overhead of computing the $|\Delta G| + 1$ instances of the all-pairs shortest paths.) Then, we generate the constraints for each (i, p) pair as presented in Figure 6.5.

Step 2 runs once (on the original graph) and step $3(b)$ runs $|\Delta G|$ times (on each topology change), generating a constraint for each alternative to the desired egress point for that configuration. As a result, the algorithm produces $(|\Delta G|+1) \cdot (|\mathcal{E}(p)|-1)$ constraints for each pair (i, p). The size of $\mathcal{E}(p)$ is limited by the number of edge nodes that have best BGP routes for a prefix; in practice, the size is usually one, two, or three, or at most ten. Fortunately, any prefixes that have the same egress set produce the same constraints, and the same values of α and β. The number of unique egress sets

1. Identify the closest egress point in the original graph:
   $b = argmin_e\{d(G, i, e) \mid e \in \mathcal{E}(p)\}$,

2. For each $e \in \mathcal{E}(p) \setminus \{b\}$, generate the constraint
   "$\alpha(i, p, b) \cdot d(G, i, b) + \beta(i, p, b) < \alpha(i, p, e) \cdot d(G, i, e) + \beta(i, p, e)$"

3. For each $\delta \in \Delta G$

   (a) Identify the preferred egress point b':
   If $d(\delta(G), i, b) \leq T \cdot d(G, i, b)$, then $b' = b$.
   Else, $b' = argmin_e\{d(\delta(G), i, e) \mid e \in \mathcal{E}(p)\}$.

   (b) For each $e \in \mathcal{E}(p) \setminus \{b'\}$, generate the constraint
   "$\alpha(i, p, b') \cdot d(\delta(G), i, b') + \beta(i, p, b') < \alpha(i, p, e) \cdot d(\delta(G), i, e) + \beta(i, p, e)$"

Figure 6.5: Algorithm of the simulation phase.

is typically orders of magnitude less than the number of prefixes, which substantially reduces the running time of the algorithm. To reduce the complexity and number of configurable parameters, we group all routers in the same PoP into a single node; these routers typically make the same BGP routing decisions anyway, since they essentially act as one larger router. Ultimately, the running time of the algorithm is dominated by the number of topology changes in ΔG.

Optimization Phase

In the optimization phase, we compute α and β values that satisfy the constraints for each pair (i, p). In theory, *any* settings that satisfy the constraints would achieve our optimization goal. However, several practical issues drive how we set up the optimization problem:

- **Finite-precision parameter values:** The α and β values should have finite precision to be configured and stored on the routers. Since the parameter values only have meaning relative to each other, we can limit ourselves to considering integer solutions, which leads us to apply *integer* programming to solve the problem.

- **Robustness to unplanned events:** Although we optimize the parameters based on the topology changes in ΔG, the real network might experience events outside of our model. If optimizing based on ΔG results in solutions with $\alpha = 0$ for an (i, p) pair, then router i would never adapt to a change in IGP distance, however large. To increase the robustness to unplanned events, we add an extra constraint that $\alpha(i, p, e) \geq 1$ for all i, p, and e.

- **Limiting the number of unique parameter values:** To reduce the overhead of configuring and storing the α and β parameters, we prefer solutions that reduce the number of unique values. As such, we attempt to minimize an objective function that is the sum across all of the α and β values, which favors solutions with $\alpha = 1$ and $\beta = 0$, selecting different values only when necessary to satisfy the constraints.

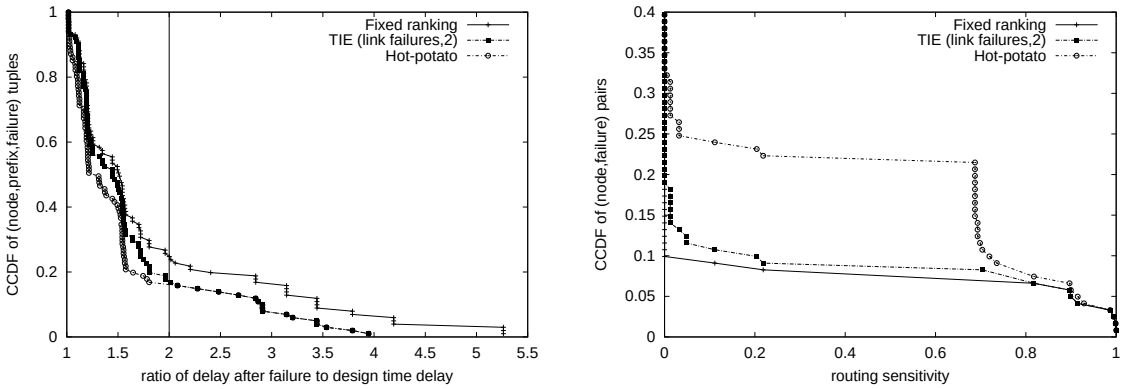
For each (i, p) pair, the simulation phase generates a set of linear inequalities and a linear objective function. Since we want our variables (α and β) to have integer values, we need to solve an integer-programming problem. We use the CPLEX [Ilo03] solver with AMPL interpreter to find the α and β values for each (i, p) pair. Although integer-programming problems are sometimes difficult to solve, our constraints are typically easy to satisfy because many constraints are identical or are subsumed by other constraints. For instance, the second constraint in Figure 6.4(b) is stricter than the first constraint (i.e., because $4\alpha_B < 6\alpha_B$). In fact, for most of the (i, p) pairs, CPLEX computes the values of α and β during a pre-processing phase that analyzes the constraints. Very few (i, p) pairs required more than three simplex iterations in the root node of

the branch-and-bound tree to identify parameters that satisfy the constraints and minimize the objective function. Still, for arbitrary topologies and topology changes, we could conceivably encounter a scenario where no parameter setting would satisfy every constraint. A scenario like this, should it arise, could be handled by an extension to the integer program to minimize the *number* of constraints that are violated. This minimization could be achieved by including an extra error term in each constraint and selecting an objective function that minimizes the total error.

6.4.3 Evaluation

We evaluate the effectiveness of TIE for achieving our goal of minimizing sensitivity to equipment failures on the Abilene network and the tier-1 ISP backbone. We obtain the network topology G and the egress sets $\{\mathcal{E}(p)\}$ as described in Section 6.3. For this problem, we set the IGP link weights to the geographic distance between the PoPs to approximate the propagation delay. We optimize TIE for two sets of topology changes ΔG (single link failures and single node failures) and three different delay thresholds T (1.5, 2, and 3).

We ran the simulation and the optimization phases on different machines because the raw measurement data could only be stored on one machine, and the CPLEX license resides on another. The simulation phase ran on a 900MHz Ultrasparc-III Copper processor of a Sun Fire 15000. This phase consumed 3.2 MB of RAM and took 0.5 and 31.1 seconds to build the constraints for all pairs (i, p) for the Abilene and ISP networks, respectively. The optimization phase ran on a 196 MHz MIPS R10000 processor on an SGI Challenge. This phase consumed just under 4 MB of RAM and took 37 seconds and 12 minutes to run for the Abilene and ISP networks, respectively. The management system selects new α and β parameters very infrequently, and this selection does not delay the routers from picking routes. Thus, 12 minutes of running time is perfectly reasonable. In addition, we expect that the optimization phase would complete



(a) Propagation delay ratio (CCDF) (b) Routing sensitivity (CCDF)

Figure 6.6: Comparison of egress-selection schemes on the Abilene network under single-node failures with TIE optimized for single-link failures and $T = 2$.

much faster if we invoke the CPLEX library directly from a C program rather than the AMPL interpreter.

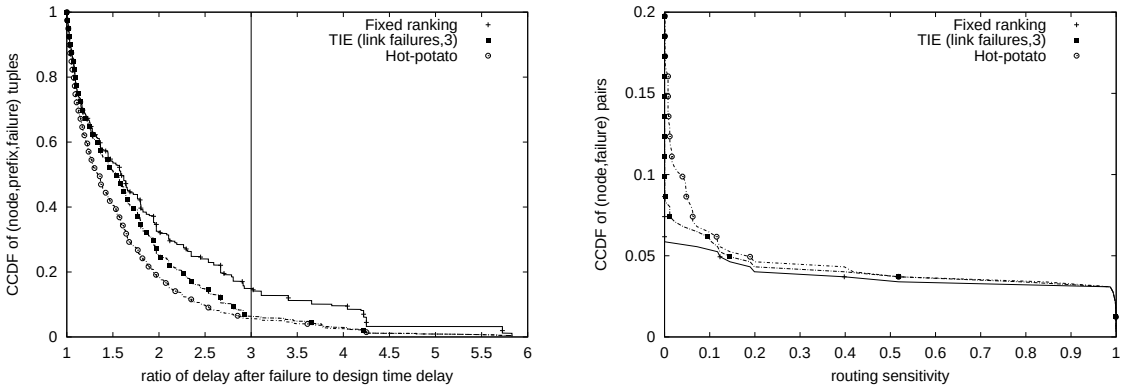
In the resulting configuration for the Abilene network, α was equal to 1 for 93% of the (i, p, e) tuples and had only four distinct values ($\alpha \in [1, 4]$); β was zero for 90% of the (i, p, e) tuples and had only three distinct values ($\beta \in \{0, 1, 3251\}$). The ISP network has a much larger number of destination prefixes and distinct egress sets, which resulted in a broader range of values for the parameters ($\alpha \in [1, 19]$ and $\beta \in \{0, 1, 3411, 4960, 5185, 5009\}$). However, the vast majority of α values (88%) were equal to one, and 69% of β values were zero. The small number of distinct values for the parameters, and the large number of $\alpha(i, p, e) = 1$ and $\beta(i, p, e) = 0$, help reduce the overhead of configuring and storing the parameters, as discussed in more detail in Section 6.6. The fact that most (i, p) pairs have $\alpha(i, p, e) = 1$ and $\beta(i, p, e) = 0$ reveals that there are just a few points in the network that need some hysteresis to keep them from over-reacting to small IGP changes. TIE provides enough flexibility such that our management system optimization routine can find which knobs to tweak to achieve the desired policy, but there is no need to tune all the knobs at the same time.

After generating the values of $\alpha(i, p, e)$ and $\beta(i, p, e)$ for each one of these scenarios, we simulate the behavior of each network with this configuration. For comparison, we also simulate the behavior of the network using hot-potato routing (by setting $\alpha(i, p, e) = 1$ and $\beta(i, p, e) = 0$ for all (i, p, e)), and the fixed ranking egress selection (by setting $\alpha(i, p, e) = 0$ for all (i, p, e), and $\beta(i, p, e) = d(G, i, b(G, i, p))$). We simulate the behavior of these egress-selection policies under the set of all single-link failures and the set of all single-node failures. For conciseness, we only present the results for single-node failures. The results for the other instances lead to the same conclusions. We compare the three mechanisms using two metrics:

- **Delay ratio:** For each (i, p, δ) we compute the delay for i to reach the best egress point for p after the topology change δ ($d(\delta(G), i, b(\delta(G), i, p))$), and divide it by the delay to reach the best egress in the original topology ($d(G, i, b(G, i, p))$).

- **Routing sensitivity:** For each (i, δ) the routing sensitivity reprints the fraction of prefixes at i that change egress point after a topology change δ. This metric is the routing-shift function ($\mathcal{H}^{RM}$) defined in Chapter 5 and represents the fraction of a router's BGP table that changes egress points after an intradomain routing change.

Figure 6.6(a) presents the complementary cumulative distribution function (CCDF) of the delay ratio for the Abilene network. A delay ratio equal to one means that the delay after the failure is the same as the delay in the original network. Many of the node failures do not affect the path between an ingress node and a best egress node for a prefix. Therefore, we omit all values that had a delay ratio of one. Given that the link weights are set according to geographic distance, the delay ratio achieved by hot-potato routing represents the smallest feasible delay ratio. Fixed ranking represents the delay to reach the old egress point after the failure. In this plot, we present the results for TIE optimized for single-link failures and $T = 2$, and evaluate the schemes against single-node failures. The results of TIE optimized for single-node failures were very similar (in fact most of the values of α and β were the same).



(a) Propagation delay ratio (CCDF) (b) Routing sensitivity (CCDF)

Figure 6.7: Comparison of egress-selection schemes on the ISP network under single-node failures for TIE optimized for single-link failures and $T = 3$.

Despite being optimized for a different set of topology changes, TIE still behaves according to the original goal. TIE exceeds the delay threshold of 2 for only 20% of the (i, p, δ), and hot-potato routing also exceeds the threshold in each of these cases. Fixing the ranking of egress points leads to delays that are higher than the delay achieved by TIE in the majority of instances. Whenever the fixed-ranking scheme lies below the threshold of 2, TIE is below it as well. When the fixed-ranking scheme exceeds the threshold, TIE shifts to an egress point that is at or below the threshold. This behavior is the reason why the TIE curve lies *below* the fixed-ranking curve for delay ratios under 2.

Below the threshold of 2, TIE has higher delay than hot-potato routing in exchange for lower sensitivity values as shown in Figure 6.6(b). This graph plots the CCDF of routing sensitivity for all (i, δ) pairs. Fixing the ranking of egress points has the lowest sensitivity. In fact, the fixed-ranking scheme has a non-zero sensitivity only when the best egress point fails, forcing even this scheme to change to the second-ranked egress point (i.e., the one that was second-closest at the initial topology). The TIE curve follows the fixed ranking for most points. TIE only experiences egress changes when

they are unavoidable. The gap between the hot-potato and the TIE curve—around 15% of the (i, δ) pairs—represents the scenarios for which egress-selection disruptions could be avoided without violating the delay threshold.

Although we observe similar behavior in the results for the large ISP network (presented in Figures 6.7(a) and 6.7(b)), the gap between the curves is not as large as for the Abilene network. In this case, we optimize TIE for single-link failures with a delay threshold $T = 3$. The ISP network has many more choices of egress points per prefixes than the Abilene network. Therefore, the delay to reach the closest egress point in the original topology is likely to be very small, and setting the threshold to three times this delay still gives reasonably short delays. This network also has more path diversity than the Abilene network. In a more diverse graph, it is more likely that there is still a low-delay path to the initial egress point, even after the failure. Contrasting the delay ratio and routing sensitivity of the two networks illustrates that there is not a single policy that fits all networks. Compared to the Abilene network, the ISP network could safely put more emphasis on setting the β values, because its rich connectivity makes it unlikely that equipment failures would lead to significant changes in the IGP distance between a pair of routers. The TIE mechanism is flexible enough to accommodate both of these networks.

In this section, we assume that the egress set for each destination prefix is stable when determining the values of α and β. Our evaluation shows that even when an egress node is removed from the egress set, TIE behaves as expected. We can extend the formulation of this problem to find solutions that are robust to egress-set changes. For instance, we can configure TIE to react slowly to the announcement of new routes (i.e., additions to the egress set) by setting the values of $\alpha(\cdot, p, e)$ and $\beta(\cdot, p, e)$ to be very high for all $e \notin \mathcal{E}(p)$. We can also extend our notion of topology changes δ to include changes to the egress sets.

6.5 Traffic Engineering

This section demonstrates the expressiveness of TIE for doing traffic engineering. We propose a management system that solves an optimization problem that balances link utilization on the network only by selecting the appropriate egress point for each pair (i, p) (i.e., by setting the values of $\beta(i, p, e)$). This approach is in contrast with the common practice of optimizing link utilization by either tweaking IGP link weights or BGP policies. After defining the optimization problem and presenting our solution, we evaluate our solution by comparing the link utilizations achieved using TIE to that using the current network configuration.

6.5.1 Problem Definition: Balancing Link Utilization

Traffic engineering—adapting the flow of traffic to the prevailing network conditions—is a common task that can be performed in several ways. Traffic engineering considers a network topology (G) with the capacity of each link ($c(\ell)$), and the traffic demands $\mathcal{M}(i, p)$ (i.e., the volume of traffic to destination prefix p that enters the network at ingress router i), as summarized in Table 6.4. The effects of the IGP weights on the intradomain paths can be represented by the matrix $R(i, e, \ell)$, which captures the fraction of traffic from router i to router e that traverses link ℓ. If the network has one shortest path between i and e, $R(i, e, \ell)$ is one for any link ℓ on that path, or zero otherwise; if multiple shortest paths exist, $R(i, e, \ell)$ may be fractional. The flow of traffic also depends on the egress set $\mathcal{E}(p)$ and the egress point $b(i, p)$ that router i uses to reach prefix p.

Traffic engineering involves tuning the network configuration to minimize some function of the load on the links. The load $t(\ell)$ on link ℓ can be determined as follows:

Link capacity	$c(\ell)$, for $\ell \in L$
Traffic demand	$\mathcal{M}(i,p)$ for $i \in N$, $p \in P$
Fraction of traffic from i to e that traverses ℓ	$R(i,e,\ell)$, for $i,e \in N$, $\ell \in L$
Egress selection	$b(i,p) \in \mathcal{E}(p)$ for $i \in N$, $p \in P$
Link traffic load	$t(\ell)$ for $\ell \in L$
Link utilization	$u(\ell) = t(\ell)/c(\ell)$, $\ell \in L$
Multicommodity flow path	$\tau(i,e,p) \subset G$
Decision variable	$x(i,e,p) \in \{0,1\}$
Link congestion penalty	$\phi(u(\ell))$, $\ell \in L$
Objective function	$\Phi = \sum_{\ell \in L} \phi(u(\ell))$

Table 6.4: Notation for the traffic-engineering problem.

$$t(\ell) = \sum_{i \in N} \sum_{\substack{p \in P, \\ b(i,p) = e, \\ e \in \mathcal{E}(p)}} \mathcal{M}(i,p) \cdot R(i,e,\ell)$$

and the resulting link utilization is $u(\ell) = t(\ell)/c(\ell)$. The common approach to traffic engineering is to formulate an optimization problem that minimizes an objective function that penalizes solutions in terms of the load they place on each link. In our work, we consider the function $\phi(u(\ell))$ in Figure 6.8 that increasingly penalizes loads as they near or pass the link's capacity. This piecewise-linear function can be expressed by the equation

$$\phi(u(\ell)) = \begin{cases} u(\ell), & u(\ell) \in [0, 1/3) \\ 3 \cdot u(\ell) - 2/3, & u(\ell) \in [1/3, 2/3), \\ 10 \cdot u(\ell) - 16/3, & u(\ell) \in [2/3, 9/10), \\ 70 \cdot u(\ell) - 178/3, & u(\ell) \in [9/10, 1), \\ 500 \cdot u(\ell) - 1468/3, & u(\ell) \in [1, 11/10), \\ 5000 \cdot u(\ell) - 16318/3, & u(\ell) \in [11/10, \infty) \end{cases} \tag{6.1}$$

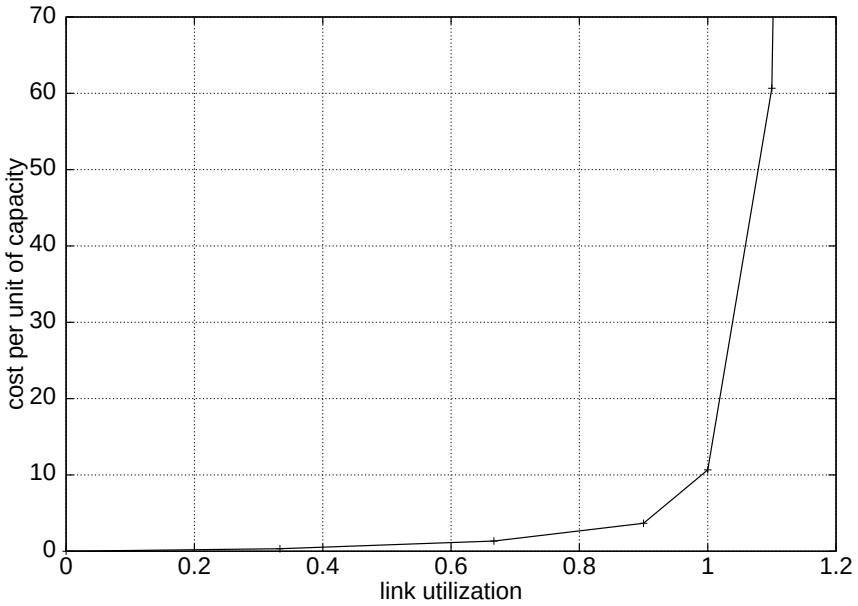


Figure 6.8: Piecewise-linear penalty function $\phi(u(\ell))$ versus link utilization.

that was introduced in [FT04] and used in several other traffic-engineering studies. The network-wide objective function Φ is the sum of the link penalties—i.e., $\Phi = \sum_{\ell \in L} \phi(u(\ell))$.

Network administrators can minimize the objective function by changing the intradomain paths ($R(i, e, \ell)$), interdomain routes ($\mathcal{E}(p)$), or the egress-point selection ($b(i, p)$). Tuning the IGP link weights (to influence the intradomain paths) and the BGP policies (to influence the interdomain routes) leads to NP-complete optimization problems [FT00, FT02, LORS01, Rex05]. The computational intractability of these problems forces the use of local-search techniques that repeatedly evaluate parameter settings in the hope of finding a good solution. Although local-search heuristics often produce good parameter values [FT00, NSTD03], the solutions are not optimal and are not guaranteed to have performance that is close to optimal. In addition, the solutions require changing the IGP weights or BGP policies, which triggers routing-protocol convergence and leads to transient disruptions. In contrast, using TIE to control the egress-point selections $b(i, p)$ leads to a simpler optimization problem that does not require changes to

the routing-protocol configuration. Since we are simply selecting among existing paths and not changing the configuration of routing protocols, our approach does not trigger routing convergence.

TIE also provides an alternative to deploying a load-sensitive routing protocol, such as the traffic-engineering extensions to OSPF and IS-IS [KKY03, Smi04, Awd99]. Load-sensitive routing leads to higher protocol overhead and can sometimes introduce instability. More recent work [KK05] solves this instability problem by balancing load over a set of pre-defined paths between ingress and egress. However, none of these proposals explicitly addresses the problem of egress-point selection, making it appealing to implement TIE even in networks that already support load-sensitive routing.

Previous research considered an optimization problem similar to the one we study here. The work in [BRS03] focused on selecting egress points such that traffic loads do not exceed the egress-point capacities, with the secondary objective of minimizing the total distance traveled by the traffic. In contrast, we formulate an optimization problem that minimizes congestion over the links in the network, using the objective function used in earlier traffic-engineering studies [FT04].

6.5.2 Solving the Traffic-Engineering Problem with TIE

Traffic engineering with TIE involves assigning each (i, p) pair to an egress point $b(i, p) \in \mathcal{E}(p)$ in a way that minimizes the objective function Φ. A solution can be realized by setting $\beta(i, p, b(i, p))$ to a low value, while setting $\beta(i, p, e)$ to a high value for all $e \neq b(i, p)$, and all α values to zero. In contrast to the fixed-ranking scheme in Section 6.1.2, we allow a router's ranking of egress points to differ across the prefixes. In practice, we envision solving richer optimization problems that consider robustness to changes in the network topology G, the egress sets $\mathcal{E}(p)$, and the traffic demands $\mathcal{M}(i, p)$, which would lead to solutions that assign values to both α and β. In this thesis, we focus on fixed topology, egress sets, and traffic demands to illustrate how TIE

provides the flexibility needed to balance load across the links.

We formulate the egress-selection problem as a *path-based* multicommodity-flow problem that accounts for the constraints that the matrix $R(i, e, \ell)$ imposes on the flow of traffic. For a router i and prefix p, we consider the topology $\tau(i, e, p)$ induced by the links $\ell \in L$ for which $R(i, e, \ell) > 0$. All links in the graph $\tau(i, e, p)$ can be used to route traffic from i to p through the egress point $e \in \mathcal{E}(p)$. We call τ a path in the multicommodity-flow formulation. We represent the actual routing of the traffic from i to p by a $(0, 1)$-decision variable $x(i, e, p)$, such that $x(i, e, p) = 1$ if and only if the path $\tau(i, e, p)$ is selected to send traffic from i to p. The choice of a path τ determines the egress point $e \in \mathcal{E}(p)$ selected. For all pairs (i, p), the egress-selection problem requires that a single egress point $e \in \mathcal{E}(p)$ be chosen. We express this requirement by the following equation:

$$\sum_{e \in \mathcal{E}(p)} x(i, e, p) = 1.$$

The contribution of the traffic going from i to p to the load on link ℓ is the product of the traffic demand $\mathcal{M}(i, p)$, the matrix element $R(i, e, \ell)$, and the decision variable $x(i, e, p)$. The total load on a link is the sum of all the contributions, i.e.

$$t(\ell) = \sum_{i \in N} \sum_{p \in P} \sum_{e \in \mathcal{E}(p)} \mathcal{M}(i, p) \cdot R(i, e, \ell) \cdot x(i, e, p).$$

A *piecewise-linear* integer-programming formulation for the single egress-selection problem is to minimize the objective function $\Phi = \sum_{\ell \in L} \phi(u(\ell))$ such that the $(0, 1)$-decision variables $x(i, e, p)$ sum to 1 for each (i, p) pair. Defining $\phi(u(\ell))$ to be a linear variable and applying a standard transformation results in the *linear* integer-programming formulation:

$$\min \sum_{\ell \in L} \phi(u(\ell))$$

s.t.

$$u(\ell) = \Big(\sum_{i \in N} \sum_{p \in P} \sum_{e \in \mathcal{E}(p)} \mathcal{M}(i,p) \cdot R(i,e,l) \cdot x(i,e,p) \Big) / c(\ell), \ \forall l \in L,$$

$$\sum_{e \in \mathcal{E}(p)} x(i,e,p) = 1, \ \forall i \in N, p \in P,$$

$$\phi(u(\ell)) \geq u(\ell), \ \forall l \in L,$$

$$\phi(u(\ell)) \geq 3 \cdot u(\ell) - 2/3, \ \forall l \in L,$$

$$\phi(u(\ell)) \geq 10 \cdot u(\ell) - 16/3, \ \forall l \in L,$$

$$\phi(u(\ell)) \geq 70 \cdot u(\ell) - 178/3, \ \forall l \in L,$$

$$\phi(u(\ell)) \geq 500 \cdot u(\ell) - 1468/3, \ \forall l \in L,$$

$$\phi(u(\ell)) \geq 5000 \cdot u(\ell) - 16318/3, \ \forall l \in L,$$

$$x(i,e,p) \in \{0,1\}, \ \forall i \in N, p \in P, e \in \mathcal{E}(p),$$

$$\phi(u(\ell)) \geq 0, \ \forall l \in L.$$

However, in general, this integer multicommodity-flow problem is intractable. Instead, we consider its linear-programming relaxation obtained by relaxing the integrality constraints $x(i,e,p) \in \{0,1\}$ to simply $x(i,e,p) \geq 0$. For both networks we consider, the CPLEX solver produced solutions with only integer values of $x(i,e,p)$, allowing us to configure the $\beta(i,p,e)$ values to pick the single egress point $b(i,p)$ for each (i,p) pair. For situations where the solution of the linear-programming relaxation is fractional, applying a simple heuristic based on randomized rounding can produce a valid egress selection. For each pair (i,p) with fractional $x(i,e,p)$ values, egress point $e \in \mathcal{E}(p)$ is selected with probability $x(i,e,p)$. The algorithm applies randomized rounding repeatedly and outputs the best solution found.

6.5.3 Evaluation

We evaluate the link utilization achieved by TIE on both the Abilene and ISP networks. We obtained the network topology G, the egress sets $\{\mathcal{E}(p)\}$, and the traffic demands $\mathcal{M}(i,p)$, as explained in Section 6.3. We aggregate all traffic from an ingress i to all destination prefixes p that share the same egress set $\mathcal{E}(p)$ to build the ingress to egress set traffic demand $\mathcal{M}(i,E)$ for each unique egress set E. For this problem, we use the IGP link weights as configured in each network. The CPLEX solver took 0.1 and 1.5 seconds to run on the 196 MHz MIPS R10000 processor for the Abilene and ISP networks, respectively. The current network IGP configuration is set to achieve good link utilization assuming that the egress-selection mechanism is hot-potato routing. Therefore, we compare the utilization achieved using TIE with that achieved by hot-potato routing.

Table 6.5 presents the value of the objective function Φ for both topologies under both egress-selection policies. TIE's flexibility in balancing load allows us to find an optimal solution for both networks using the linear-programming relaxation. The solution using hot-potato routing is 40% worse than that found using TIE for the ISP network. Hot-potato routing has a congestion function close to TIE for the Abilene network. However, even though the Abilene network is significantly under-utilized, TIE does offer some (admittedly modest) improvements to the objective function.

	Abilene Network	ISP Network
Hot-potato routing	0.4513510071	8.990353677
TIE	0.4425879808	5.557480707

Table 6.5: Comparison of the network congestion function Φ between hot-potato routing and TIE.

Figure 6.9 shows the ratio of link utilization between hot-potato routing and TIE, for the ten most heavily-loaded links under hot-potato routing; link number 1 is the most utilized link and number 10 is the tenth most utilized. The TIE solution reduces

the utilization of the most utilized link by 40.9%. Although TIE increases the load on some links (as illustrated by link 8 in the figure), our solution reduces the utilization of two-thirds of the links, and the most utilized link in the TIE solution has 26.3% less utilization than the most utilized link under hot-potato routing.

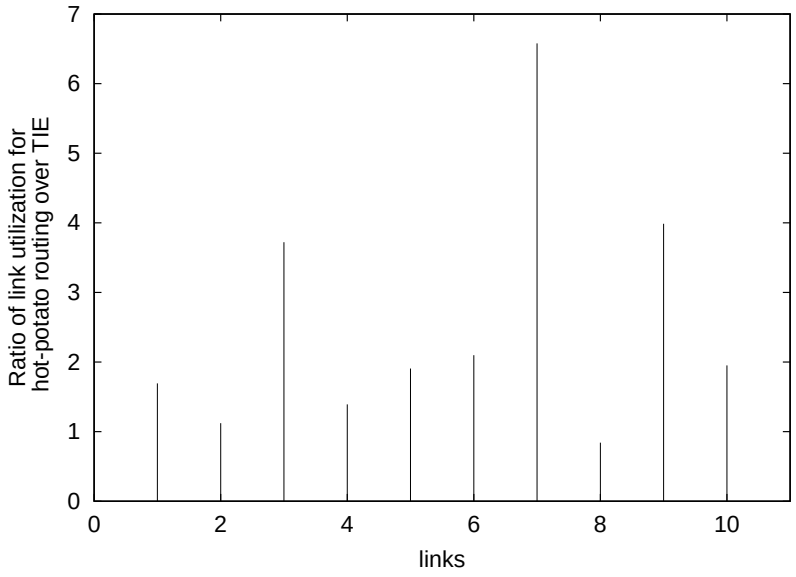


Figure 6.9: Comparison of link utilization with hot-potato routing and TIE.

In our ongoing work, we plan to compare the TIE solution with the loose lower bound achieved by multicommodity flow with no restrictions on using valid IGP paths. We also want to compare this solution with that achieved by using other traffic-engineering mechanisms: (i) heuristics for IGP link-weight optimization; (ii) heuristics for setting local-preference values in BGP import policies; and (iii) egress-point optimization where each router i is forced to have a single ranking of egress points across all destination prefixes, as in Section 6.1.2. These comparisons will help us understand how much of the performance benefit of TIE comes from the decoupling of egress selection from the IGP weights versus the ability to exert fine-grain control over the ranking of egress points.

6.5.4 Extensions

In this section, we assume that each router i can select any $e \in \mathcal{E}(p)$ for each destination prefix p. However, this could conceivably lead to long propagation delays if i selects a far-away egress point, or to unnecessary BGP update messages to neighboring domains. We can address these concerns simply by removing certain egress points from consideration if they have high propagation delay or a BGP route with a different AS path. For instance, egresses where $d(G, i, e)$ exceeds a threshold could be removed from consideration for router i, or we could consider only the egress points that have BGP routes with the same AS path. Our solution can also treat destination prefixes for sensitive applications (such as VoIP) separately. For instance, the egress selection for such prefixes can be done to minimize sensitivity and delay as discussed in Section 6.4, and the demands to these prefixes considered as immutable background load for the traffic-engineering problem.

The traffic-engineering optimization problem as defined in this section only considers the utilization of internal links. A natural extension is to use TIE to *balance outbound load on the edge links*. We can formulate this problem by adding an artificial node for each destination prefix p, with each peering link connecting to it, and solve it using the same methodology presented here. In addition, our traffic-engineering optimization problem currently does not set the values of α, which prevents the egress selection to automatically adapt to changes in the network topology. We can combine our methodology for solving the problem presented in Section 6.4 with the one presented here to find a solution to the *robust traffic-engineering* problem. In steps 1 and 3(a) in Figure 6.5, instead of identifying the best egress point according to the shortest distance, we can achieve robust traffic engineering by selecting the best egress according to the solution of the path-based multicommodity-flow problem specified in Section 6.5.2. TIE can also be configured before *planned maintenance activities* to ensure low link utilizations during the event. In this case, the topology change δ is known in advance,

so the network administrators can compute the optimal egress selection in the modified topology $\delta(G)$ and adjust α and β to achieve the desired traffic-engineering goal.

6.6 Implementation Issues

An AS can deploy the TIE mechanism without changing the intradomain or interdomain routing protocols, and without the cooperation of other domains. In this section, we first describe how to ensure that each router can apply TIE independently of other routers in the AS. Next we discuss how to configure the α and β parameters and how a router applies the TIE mechanism to select a BGP route for each destination prefix. Then, we discuss how moving the responsibility for BGP path selection from the routers to separate servers [FBR+04, BUQ04] would make it possible to implement our TIE scheme without *any* modification to the decision logic running on the routers.

6.6.1 Allowing Independent Decisions at Each Node

Throughout the chapter, we have assumed that each node applies the TIE mechanism to select a single best route from the set of equally-good BGP routes chosen by the border routers. In a network with a "full mesh" internal BGP (iBGP) configuration, each router learns these routes directly from the border routers. However, large networks typically employ route reflectors to overcome the scaling problems of having an iBGP session for each pair of routers. A route reflector runs the BGP decision process and propagates a single best route to its clients; as a result, the clients may choose a different best route than they would with all of the options at their disposal. The way route reflectors affect the BGP decisions of their clients leads to a variety of operational problems, such as protocol oscillation and forwarding loops [GW02b, BRO+02, MGWR02]. An appealing way to avoid these problems, while retaining most of the scalability advantages, is to have the route reflectors forward *all* of the equally-good BGP routes to their clients [BRO+02]. This enhancement to route reflectors would allow each router

in the AS to apply the TIE mechanism based on a complete view of the egress set for each destination prefix.

Consider the common scenario with a full mesh of top-level route reflectors, with one or more route reflectors in each PoP. In this scenario, we recommend applying the TIE mechanism only on the route reflectors to allow decisions based on a complete view of the BGP routes. The client routers (i.e., other routers in the same PoP) would inherit the choice made by their common route reflector. This approach has the added advantage that only the route reflectors would need to be upgraded to implement the TIE mechanism.

The TIE mechanism also relies on the underlying network to forward data packets from the ingress router to the chosen egress point. However, the routers along the forwarding path do not necessarily select the same egress point, depending on how their α and β parameters are configured. This problem does not arise in hot-potato routing because each router selects the closest egress point, which ensures that the routers along the shortest path have chosen the same egress point. Rather than constraining the way α and β are set on different routers, we advocate that the network employ some form of lightweight tunneling to direct traffic over the shortest IGP path(s) from the ingress point to the egress point. For example, the ingress router could encapsulate each data packet in an IP packet where the destination corresponds to the chosen egress router. Alternatively, the network may employ MPLS [RVC01, DR00] to create label-switched paths (LSPs) between all ingress-egress pairs, as discussed earlier in Section 6.1.2. Tunneling IP packets over the underlying IGP paths is a common usage of MPLS since it obviates the need for interior routers to speak BGP or have a large forwarding table, while also allowing the network to forward VPN and non-IP traffic.

6.6.2 Configuring and Applying TIE in Routers

Using the TIE mechanism requires configuring the routers with the values of α and β selected by the optimization routine. Rather than configuring these values by hand, we envision that a network-management system would connect to each router to set or modify the parameters. Still, configuring a large number of values may introduce significant overhead and delay. In the worst case, each router would need to be configured with two integer values for every destination prefix and edge router. For a network with 500 edge routers and 150,000 destination prefixes, this would require configuring 75 billion parameters (i.e., $500 \cdot 500 \cdot 2 \cdot 150,000$), which is clearly excessive. Fortunately, a router often has the same values of α and β across many destination prefixes and egress points. To capitalize on this observation, the TIE mechanism could have default values of $\alpha = 1$ and $\beta = 0$ (corresponding to hot-potato routing) for each prefix, allowing the management system to specify only the parameters that differ from these values. For example, in Section 6.4 only 10% of the β values were non-zero for the tier-1 ISP backbone, which would reduce the configuration overhead by an order of magnitude.

Another way to reduce the overhead is to assign α and β at a coarser granularity than individual routers and destination prefixes. For example, the parameters could be defined for PoPs rather than routers, particularly if TIE is implemented only at the route reflector(s) in each PoP. If the 500-router network has (say) 25 PoPs, the number of parameters would drop by a factor of 400 (i.e., 25 PoPs would be configured with two parameters per prefix for 25 egress PoPs). In addition, the parameters could be based on the destination AS (i.e., the origin AS that initially announced the BGP route), rather than the destination prefix. If the Internet has (say) 20,000 ASes and 150,000 prefixes, this would reduce the number of parameters by an additional factor of 7.5. Together, these two optimizations would reduce the number of parameters by a factor of 3000, from 75 billion down to 25 million across all the routers in the network. This number

of parameters seems acceptable particularly if the management system need only specify exceptions to the default α and β values. Further reductions can be achieved by associating α and β values with the next-hop AS or other route attributes.

When α and β are not associated directly with particular prefixes and egress routers, the ingress router needs some way to know which parameters to use in selecting a BGP route for a prefix. The BGP *community* attribute [CTL96] provides an effective way to communicate which parameters should be used. For example, the border routers could be configured to tag each BGP advertisement with a unique community value that identifies the PoP. Another community could be used to identify the origin AS or next-hop AS associated with the advertisement. Upon receiving these tagged routes via internal BGP (iBGP), a router can use these community values to index into a table that stores the α and β values. Using BGP communities in this way is quite common. For example, policy-based accounting uses community attributes to determine which prefixes should have their traffic measured together by a single counter [Cisc].

Once the router knows which α and β values to use, the router can compute the metric m based on these parameters and the IGP distance to the egress router. Rather than applying the traditional IGP tie-breaking step, the router can implement a modified BGP decision process that uses the m metric to select the route with the most-preferred egress point. Ultimately, the TIE mechanism requires only a change in one step of the BGP decision process implemented on the routers, rather than any protocol modifications. We note that router vendors already provide features that allow network administrators to modify the operation of the BGP decision process [Cisb], which significantly reduces the barrier to deploying TIE.

6.6.3 Applying TIE in a Separate Path-Selection Platform

Rather than modifying the BGP decision process implemented on the routers, an AS could move the entire responsibility for BGP path selection to a separate software

platform, as proposed in [FBR+04, BUQ04]. In this setting, dedicated servers receive the eBGP advertisements and run decision logic to select BGP routes on behalf of the routers in the AS. The servers use iBGP sessions to send each router a customized routing decision for each prefix, essentially overriding the influence of the BGP decision process running on the routers.

These servers could implement the TIE mechanism for selecting the routes in real time, and might also run the offline optimization routines that set the α and β parameters; this would allow the parameters to exist only on the servers, rather than in the routers or other management systems. Even though the servers could conceivably implement any decision logic, in practice they need some separation of functionality between the real-time adaptation to network events and the longer-term optimization of the path-selection process based on network-wide goals. TIE provides a way to achieve that separation.

6.7 Summary

In this chapter, we propose a new mechanism for selecting egress points that is configurable to accommodate the requirements of different networks and applications. TIE avoids the disruptions caused by hot-potato routing changes while supporting diverse network-wide objectives such as traffic engineering and maintenance planning. TIE is simple enough for routers to adapt in real time to network events, and yet is much more amenable to optimization than today's routing protocols. In addition, TIE can be deployed in an AS without changing the intradomain or interdomain routing protocols, and without the cooperation of other domains. Our experiments for two network-management problems, using data from two backbone networks, demonstrate the effectiveness of our new mechanism and the ease of applying conventional optimization techniques to determine the best settings for the tunable parameters.

Chapter 7

Conclusion

As the Internet becomes an increasingly important part of the world's communication infrastructure, we must consider robustness as a first-order objective in the design and operation of network protocols. In light of the growing pressure to provide good, predictable communication performance for applications such as voice-over-IP, online gaming, and business transactions, transient disruptions (i.e., during routing changes) and persistent congestion (i.e., when the routing does not match the prevailing traffic) can no longer be tolerated.

In this thesis, we show that many large disruptions stem from hot-potato routing changes, i.e., internal routing changes that affect the interdomain routing decisions in BGP. We propose two approaches for addressing this issue that give more control for network administrators to minimize disruptions: a model of network sensitivity and a mechanism for egress point selection called Tunable Interdomain Egress selection (TIE). Our network sensitivity model allows network administrators to engineer their network using current router implementation. The TIE mechanism for selecting egress points allows more flexible egress point selection policies that can be optimized to achieve many traffic engineering goals.

This thesis makes the following contributions:

- **Methodology for identifying hot-potato routing changes**. Our algorithm to correlate intradomain and interdomain routing data to identify hot-potato routing changes was the first work on joint analysis of measurement data from both protocols. This algorithm is being used at a tier-1 ISP network to identify the impact of maintenance activities and failures on interdomain routes.

- **Showed that internal events can have big impact on interdomain routing dynamics and on other networks**. We characterized the dynamics of BGP updates triggered by internal routing changes in an operational network. This characterization showed that some intradomain routing changes cause more than half of a router's BGP table to change egress points. This massive egress shift impacts not only the domain where the change happened, but also traffic and routing in other domains. In fact, these routing changes are responsible for the largest traffic shifts. Our work also showed that the network can take over a minute to converge after an internal change that affects BGP routes. This convergence time is an order of magnitude larger than normal intradomain convergences times. This analysis prompted a major router vendor to reimplement the interaction between the two routing protocols.

- **Model of network sensitivity to internal routing changes**. After our empirical analysis of the interaction between intradomain and interdomain routing, we developed methods for characterizing network sensitivity to intradomain routing changes to ultimately improve network robustness. First, we propose and describe an analytic model of the interaction between intra and interdomain routing and its impact on both the control and data planes of an ISP network. Based on this model, we define a set of metrics for describing a network's sensitivity to intradomain routing perturbations. We study control plane sensitivity of a large AS of a tier-1 ISP to link and router failures. Our model has later been used for selecting the placement of probe machines to study network instabilities, and to analyze the

sensitivity of the European research network (GEANT) [Uhl05].

- **Mechanism for selecting egress points**. Our sensitivity model focused on improving network design given routing protocols as they exist today. We then propose a new egress selection mechanism called TIE. TIE replaces, at each router, the step of the BGP decision logic that selects egress points. TIE is simple enough to be computed in real time by routers and expressive for a higher-level management system to optimize to achieve a wide range of network design and engineering goals. We show that TIE can be used to solve two practical problems: minimize sensitivity to equipment failures and traffic engineering.

In summary, this thesis shows that the interaction between the two-tiers of the routing system can cause network service disruptions that jeopardize network robustness. Our network sensitivity model allows network administrators to gain knowledge of the vulnerabilities of the network, which can be used to engineer more robust networks. Changing the BGP decision in the routers to implement the TIE mechanism gives network administrators even more fine-grained control over egress point selection. As a result, ISPs can meet robustness and performance goals required by a wide variety of applications.

Bibliography

[Abi] Abilene Backbone Network. `http://abilene.internet2.edu/`.

[ACBD04] Sharad Agarwal, Chen-Nee Chuah, Supratik Bhattacharyya, and Christophe Diot. Impact of BGP Dynamics on Intra-Domain Traffic. In *Proc. ACM SIGMETRICS*, June 2004.

[AJY00] Cengiz Alaettinoglu, Van Jacobson, and Haobo Yu. Toward Milli-Second IGP Convergence, November 2000. Expired Internet Draft, draft-alaettinoglu-isis-convergence-00.txt.

[AMS+03] Aditya Akella, Bruce Maggs, Srinivasan Seshan, Aman Shaikh, and Ramesh Sitaraman. A measurement-based analysis of multi-homing. In *Proc. ACM SIGCOMM*, August 2003.

[Awd99] D.O. Awduche. MPLS and Traffic Engineering in IP Networks. *IEEE Communication Magazine*, December 1999.

[BID02] C. Boutremans, G. Iannacconne, and C. Diot. Impact of Link Failures on VoIP Performance. In *Proc. of NOSSDAV workshop*. ACM Press, May 2002.

[BRO+02] Anindya Basu, April Rasala, C.-H. Luke Ong, F. Bruce Shepherd, and Gordon Wilfong. Route Oscillations in I-BGP with Route Reflection. In *Proc. ACM SIGCOMM*, August 2002.

[BRRT03] L.S. Buriol, M.G.C. Resende, C.C. Ribeiro, and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. Technical Report TD-5NTN5G, AT&T Labs Research, 2003.

[BRS03] T.C. Bressoud, R. Rastogi, and M.A. Smith. Optimal configuration of BGP route selection. In *Proc. IEEE INFOCOM*, 2003.

[BUQ04] O. Bonaventure, S. Uhlig, and B. Quoitin. The Case for More Versatile BGP Route Reflectors. Internet Draft draft-bonaventure-bgp-route-reflectors-00.txt, July 2004.

[Cal90] R. Callon. Use of OSI IS–IS for Routing in TCP/IP and Dual Environments. RFC1195, December 1990.

[CD97] Surajit Chaudhuri and Umesh Dayal. An Overview of Data Warehousing and OLAP Technology. *ACM SIGMOD Record*, 26(1), March 1997.

[CDWY00] J. Cao, D. Davis, S. Vander Wiel, and B. Yu. Time-varying network tomography. *J. American Statistical Association*, December 2000.

[Cisa] Cisco. BGP Best Path Selection Algorithm. `http://www.cisco.com/warp/public/459/25.shtml`.

[Cisb] Cisco. BGP cost community. `http://www.cisco.com/en/US/products/sw/iosswrel/ps5207/products_feature%_guide09186a00801a7f74.html`.

[Cisc] Cisco. BGP policy accounting. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newf%t/122t/122t13/ft_bgppa.htm`.

[Cisd] Cisco. Configure Router Calculation Timers. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/%np1_c/1cprt1/1cospf.html#xtocid2712621`.

[Cise] Cisco. Configuring OSPF. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr%/ip_c/ipcprt2/1cdospf.htm`.

[Cisf] Cisco. Sampled Netflow. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newf%t/120limit/120s/120s11/12s_sanf.htm`.

[Cisg] Cisco. Understanding BGP Processes on Cisco. `http://www.cisco.com/warp/public/459/highcpu-bgp.html#topic1`.

[Cla88] David D. Clark. The Design Philosophy of the DARPA Internet Protocols. In *Proc. ACM SIGCOMM*, pages 106–114, August 1988.

[CSK03] Matthew Caesar, Lakshminarayanan Subramanian, and Randy H. Katz. Towards localizing root causes of BGP dynamics. Technical Report CSD-03-1292, UC Berkeley, November 2003.

[CTL96] R. Chandra, P. Traina, and T. Li. BGP communities attribute. RFC 1997, August 1996.

[Dij59] E. W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik 1*, pages 269–271, 1959.

[DLT03] Nick Duffield, Carsten Lund, and Mikkel Thorup. Estimating flow distributions from sampled flow statistics. In *Proc. ACM SIGCOMM*, August 2003.

[DR00] Bruce S. Davie and Yakov Rekhter. *MPLS: Technology and Applications*. Morgan Kaufmann, May 2000.

[ERP02] M. Ericsson, M.G.C. Resende, and P.M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *Journal of Combinatorial Optimization*, 6:299–333, 2002.

[FBR03] Nick Feamster, Jay Borkenhagen, and Jennifer Rexford. Guidelines for Interdomain Traffic Engineering. *ACM SIGCOMM Computer Communication Review*, October 2003.

[FBR+04] Nick Feamster, Hari Balakrishnan, Jennifer Rexford, Aman Shaikh, and Jacobus van der Merwe. The Case for Separating Routing from Routers. In *ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2004.

[FGL+00] Anja Feldmann, Albert Greenberg, Carsten Lund, Nick Reingold, and Jennifer Rexford. NetScope: Traffic Engineering for IP Networks. *IEEE Network Magazine*, pages 11–19, March 2000.

[FGL+01] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving Traffic Demands for Operational IP Networks: Methodology and Experience. *IEEE/ACM Trans. Networking*, 9(3), June 2001.

[FMM+04] Anja Feldmann, Olaf Maennel, Z. Morley Mao, Arthur Berger, and Bruce Maggs. Locating Internet Routing Instabilities. In *Proc. ACM SIGCOMM*, September 2004.

[FRT02] Bernard Fortz, Jennifer Rexford, and Mikkel Thorup. Traffic Engineering with Traditional IP Routing Protocols. *IEEE Communication Magazine*, October 2002.

[FT00] Bernard Fortz and Mikkel Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proc. IEEE INFOCOM*, March 2000.

[FT02] Bernard Fortz and Mikkel Thorup. Optimizing OSPF/IS-IS Weights in a Changing World. *IEEE J. Selected Areas in Communications*, 20(4):756 – 767, 2002.

[FT03] B. Fortz and M. Thorup. Robust optimization of OSPF/IS-IS weights. In *Proc. International Network Optimization Conference*, pages 225–230, October 2003.

[FT04] B. Fortz and M. Thorup. Increasing internet capacity using local search. *Computational Optimization and Applications*, 29(1):13–48, 2004. Preliminary short version of this paper published as "Internet Traffic Engineering by Optimizing OSPF weights," in Proc. 19th IEEE Conf. on Computer Communications (INFOCOM 2000).

[FV00] Andrea Fumagalli and Luca Valcarenghi. IP Restoration vs. WDM Protection: Is There an Optimal Choice? *IEEE Network Magazine*, November/December 2000.

[FWR04] Nick Feamster, Jirad Winick, and Jennifer Rexford. A Model of BGP Routing for Network Engineering. In *Proc. ACM SIGMETRICS*, June 2004.

[GBLP97] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A Relational Aggregation Operator Generalizing Group-by, Cross-tabs and Subtotals. *Data Mining and Knowledge Discovery*, (1):29–53, 1997.

[GHQ95] A. Gupta, V. Harinarayan, and D. Quass. Aggregate query processing in data warehousing environments. In *VLDB*, pages 358–369, 1995.

[GQX+04] David K. Goldenberg, Lili Qiu, Haiyong Xie, Yang Richard Yang, and Yin Zhang. Optimizing cost and performance for multihoming. In *Proc. ACM SIGCOMM*, Portland,OR, September 2004.

[GR97] Ramesh Govindan and Anoop Reddy. An Analysis of Internet Inter-Domain Topology and Route Stability. In *Proc. IEEE INFOCOM*, April 1997.

[GR01] Lixin Gao and Jennifer Rexford. Stable Internet Routing without Global Coordination. *IEEE/ACM Trans. Networking*, December 2001.

[GW99] Tim Griffin and Gordon Wilfong. An Analysis of BGP Convergence Properties. In *Proc. ACM SIGCOMM*, September 1999.

[GW02a] Tim Griffin and Gordon Wilfong. An Analysis of the MED Oscillation Problem in BGP. In *Proc. International Conference on Network Protocols*, 2002.

[GW02b] Timothy G. Griffin and Gordon Wilfong. On the Correctness of IBGP Configuration. In *Proc. ACM SIGCOMM*, August 2002.

[HM01] Sam Halabi and Danny McPherson. *Internet Routing Architectures*. Cisco Press, second edition, 2001.

[HMMD02] Urs Hengartner, Sue B. Moon, Richard Mortier, and Christophe Diot. Detection and Analysis of Routing Loops in Packet Traces. In *Proc. Internet Measurement Workshop*, November 2002.

[ICBD04] Gianluca Iannaccone, Chen-Nee Chuah, Supratik Bhattacharyya, and Christophe Diot. Feasibility of IP Restoration in a Tier-1 Backbone. *IEEE Network Magazine*, March 2004.

[Ilo03] Ilog S.A. *"Ilog Cplex 9.0 User's Manual"*, October 2003.

[InCM+02] Gianluca Iannaccone, Chen nee Chuah, Richard Mortier, Supratik Bhattacharyya, and Christophe Diot. Analysis of link failures in an IP backbone. In *Proc. Internet Measurement Workshop*, November 2002.

[IPM] IP Monitoring Project. `http://ipmon.sprint.com/`.

[IPS] Route Dynamics. `http://www.ipsumnetworks.com`.

[JT03] Ramesh Johari and John Tsitsiklis. Routing and Peering in a Competitive Internet. Technical report, MIT, January 2003.

[KK05] Srikanth Kandula and Dina Katabi. TeXCP: Responsive Yet Stable Traffic Engineering. In *Proc. ACM SIGCOMM*, August 2005.

[KKY03] D. Katz, K. Kompela, and D. Yeung. Traffic Engineering (TE) Extensions to OSPF Version 2. RFC 3630, September 2003.

[LABJ01] Craig Labovitz, Abha Ahuja, A. Bose, and Farnam Jahanian. Delayed Internet Routing Convergence. *IEEE/ACM Trans. Networking*, 9(3):293–306, June 2001.

[LAJ99] Craig Labovitz, A. Ahuja, and Farnam Jahanian. Experimental Study of Internet Stability and Wide-Area Network Failures. In *Proc. International Symposium on Fault-Tolerant Computing*, June 1999.

[LCD04] A. Lakhina, M. Crovella, and C. Diot. Characterization of Network-Wide Anomalies in Traffic Flows. In *Proc. Internet Measurement Conference*, October 2004.

[LMJ98] Craig Labovitz, Rob Malan, and Farnam Jahanian. Internet Routing Instability. *IEEE/ACM Trans. Networking*, 6(5):515–558, October 1998.

[LMJ99] Craig Labovitz, Rob Malan, and Farnam Jahanian. Origins of Pathological Internet Routing Instability. In *Proc. IEEE INFOCOM*, March 1999.

[LORS01] Dean H. Lorenz, Ariel Orda, Danny Raz, and Yuval Shavitt. How good can IP routing be? Technical Report 2001-17, DIMACS, May 2001.

[LPC+04] Anukool Lakhina, Konstantina Papagiannaki, Mark Crovella, Christophe Diot, Eric Kolaczyk, and Nina Taft. Structural Analysis of Network Traffic Flows. In *Proc. ACM SIGMETRICS*, June 2004.

[MGVK02] Z. Morley Mao, Ramesh Govindan, George Varghese, and Randy Katz. Route Flap Damping Exacerbates Internet Routing Convergence. In *Proc. ACM SIGCOMM*, August 2002.

[MGWR02] Danny McPherson, Vijay Gill, Daniel Walton, and Alvaro Retana. Border gateway protocol (BGP) persistent route oscillation condition. RFC 3345, August 2002.

[Moy98] John Moy. OSPF Version 2. RFC 2328, April 1998.

[MRWK03] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy H. Katz. Towards an accurate AS-level traceroute tool. In *Proc. ACM SIGCOMM*, August 2003.

[MTSD02] A. Medina, N. Taft, K. Salamatian, and C. Diot. A Taxonomy of IP Traffic Matrix Estimation: Existing Techiniques and New Directions. In *Proc. ACM SIGCOMM*, August 2002.

[MWA04] Ratul Mahajan, David Wetherall, and Thomas Anderson. Towards coordinated interdomain traffic engineering. In *Proc. SIGCOMM Workshop on Hot Topics in Networking*, November 2004.

[NSTD03] A. Nucci, B. Schroeder, N. Taft, and C. Diot. IGP Link Weight Assignment for Transient Link Failures. In *Elsevier ITC18*, August 2003.

[Pkt] Route Explorer. `http://www.route-explorer.com/`.

[QUP+03] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure. Interdomain traffic engineering with BGP. In *IEEE Communication Magazine*, May 2003.

[Rex05] Jennifer Rexford. Route optimization in IP networks. In Panos Pardalos and Mauricio Resende, editors, *Handbook of Optimization in Telecommunications*. Kluwer Academic Publishers, 2005. To appear.

[RIP] RIPE NCC RIS. `http://www.ripe.net/ripencc/pub-services/np/ris-index.html`.

[RL95] Yakov Rekhter and Tony Li. A Border Gateway Protocol 4 (BGP-4). RFC 1771, March 1995.

[RLH04] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). Internet Draft draft-ietf-idr-bgp4-25.txt, September 2004.

[RV] Route Views Project. `http://www.routeviews.org`.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, January 2001.

[RWXZ02] Jennifer Rexford, Jia Wang, Zhen Xiao, and Yin Zhang. BGP Routing Stability of Popular Destinations. In *Proc. Internet Measurement Workshop*, November 2002.

[SARK02] Lakshminarayanan Subramanian, Sharad Agarwal, Jennifer Rexford, and Randy H. Katz. Characterizing the Internet hierarchy from multiple vantage points. In *Proc. IEEE INFOCOM*, June 2002.

[SG04] Aman Shaikh and Albert Greenberg. OSPF Monitoring: Architecture, Design and Deployment Experience. In *Proc. USENIX Symposium on Networked Systems Design and Implementation*, March 2004.

[SG05] A. Sridharan and R. Guerin. Making OSPF/IS-IS routing robust to link failures. In *Proc. Networking'2005*, Ontario, Canada, May 2005.

[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 *Proc. Internet Measurement Workshop*, November 2002.

[SMA03] Neil Spring, Ratul Mahajan, and Thomas Anderson. Quantifying the Causes of Path Inflation . In *Proc. ACM SIGCOMM*, August 2003.

[SMD03] Ashwin Sridharan, Sue B. Moon, and Christophe Diot. On the Correlation between Route Dynamics and Routing Loops. In *Proc. Internet Measurement Conference*, October 2003.

[Smi04] H. Smit. Intermediate System to Intermediate System (IS-IS) Extensions for Traffic Engineering (TE). RFC 3784, June 2004.

[TSGR04] Renata Teixeira, Aman Shaikh, Tim Griffin, and Jennifer Rexford. Dynamics of Hot-Potato Routing in IP Networks. In *Proc. ACM SIGMETRICS*, June 2004.

[Uhl04] Steve Uhlig. A multiple-objectives evolutionary perspective to interdomain traffic engineering in the internet. In *Workshop on Nature Inspired Approaches to Networks and Telecommunications (NIANT)*, Birmingham, UK, September 2004.

[Uhl05] Steve Uhlig. On the Sensitivity of Transit ASes to Internal Failures. In *Proc. of the IEEE International workshop on IP Operations and Management*, Barcelona, Spain, October 2005.

[WJR02] Jared Winick, Sugih Jamin, and Jennifer Rexford. Traffic engineering between neighboring domains. `http://www.cs.princeton.edu/~jrex/papers/interAS.pdf`, July 2002.

[WLJ03] David Watson, Craig Labovitz, and Farnam Jahanian. Experiences with Monitoring OSPF on a Regional Service Provider Network. In *Proc. International Conference on Distributed Computing Systems*, pages 204–213, May 2003.

[WZP+02] Lan Wang, Xiaoliang Zhao, Dan Pei, Randy Bush, Daniel Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. Observation and Analysis of BGP Behavior Under Stress. In *Proc. Internet Measurement Workshop*, November 2002.

[ZRLD03] Y. Zhang, M. Roughan, C. Lund, and D. Donoho. An Information-Theoretic Approach to Traffic Matrix Estimation. In *Proc. ACM SIGCOMM*, August 2003.