Efficient discovery of network topology and routing policy in the Internet

Neil Timothy Spring

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Washington

2004

Program Authorized to Offer Degree: Computer Science and Engineering

University of Washington

Abstract

Efficient discovery of network topology and routing policy in the Internet

by Neil Timothy Spring

Co-Chairs of Supervisory Committee:

Associate Professor David J. Wetherall
Department of Computer Science and Engineering

Professor Thomas E. Anderson
Department of Computer Science and Engineering

Little is known about the structure and configuration of the ISP networks that constitute the Internet. This shortage of information is a consequence of a fundamental principle of the Internet architecture: that constituent ISPs are administered independently. ISP networks connect by a narrow interface that supports the delivery of data but hides the implementation and behavior of a network from its neighbors. One consequence of this isolation between networks is that, although operators have inside information for their own networks, it is difficult to recognize and fix problems that span administrative boundaries.

In this dissertation, I focus on what can be discovered by an outsider: a user or researcher without access to privileged information. I show that the network topologies and routing policies of ISP networks can be discovered despite the narrowness of the interface between them. To do this, I develop and evaluate techniques to measure structured, router-level ISP network topologies and infer intra-domain and peering routing policies. To make these techniques efficient, I use a philosophy of choosing to collect only measurements likely to yield new information. This enables the techniques to run on a large network measurement platform, composed of hundreds of public traceroute servers, to produce an accurate result.

I applied and evaluated my techniques to map ten diverse ISP networks and characterize the routing policies of 65 ISP networks. The results are a set of ISP topologies that are several times more complete than previous maps and the first quantitative study of peering routing policy. This data highlights the diversity of ISP networks and can be used by others to better understand network operation and protocol design choices.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number Page

LIST OF TABLES

Table Number Page

ACKNOWLEDGMENTS

My gratitude to those who have helped me complete this dissertation cannot be adequately expressed here. Please accept my apologies if you find yourself unjustly missing or find your contribution inadequately credited. I am truly grateful for your assistance.

I have been fortunate to have both David Wetherall and Thomas Anderson as advisors for this work. Their research styles blend perfectly. They provided the enthusiasm, encouragement, and reassurance I knew that I needed to complete this work. As I look forward to starting as a professor on my own, I wonder just how many things they did well enough for me not to notice.

Rich Wolski, my undergraduate advisor, let me run wild with engineering and synthesized the result into research. Along the way, he introduced me to some of the things I am now excessively passionate about: Latex, Strunk and White, and research.

I was fortunate to work with Ratul Mahajan on nearly all the work presented in this dissertation. Ratul is brilliant, clever, and intense; I tried my best to keep up and stay awake so late. My other collaborators brought me their curiosity and gave me a chance to relax, teach, learn, and focus on their success: Stavan Parikh, Mira Dontcheva, David Ely, and Maya Rodrig.

My parents gave me their love and pride, never encumbering me with guidance, advice, or disappointment. Several teachers inspired me to teach and played crucial roles in directing me here: Francine Berman, Randy Borden, Mike Casey, Robert Lunsford, Mary Mauer, Keith Marzullo, Georgiana Orozco, and Jay Siegel.

Various sources provided funding for the work described here, including Intel, NSF, Cisco, and DARPA.

Neal Cardwell and Stefan Savage shared curious TCP problems with me when I was a young graduate student, leading me to a path of studying networks.

My close friends and colleagues made this work possible by encouraging my growth and keeping obstacles out of my way: Gretta Cook, Krishna Gummadi, Kurt Partridge, Rachel Pottinger, Vibha Sazawal, Marianne Shaw, Steven Wolfman, Brian Youngstrom, and Ken Yasuhara.

Several faculty provided me with their guidance about graduate school and feedback about this work, including Brian Bershad, Gaetano Boriello, Steve Gribble, Anna Karlin, Edward Lazowska, and John Zahorjan. Ken Yasuhara, Andy Collins, Richard Dunn, Krishna Gummadi, Stefan Saroiu, Gang Zhao, and the SIGCOMM reviewers provided very helpful feedback on various drafts of the papers where much of this work first appeared.

I was only able to construct the tools used here by synthesizing various free software packages. The software used in this dissertation includes tcpdump, nmap, ruby, perl, autoconf, gcc, PostgreSQL, Condor, and lpsolve. Many analyses were supported by Linux machines running Debian.

The PlanetLab test-bed [97] has made it possible for researchers like me to run ambitiously large measurements like mine with relatively little difficulty. I am thankful that the vision for PlanetLab defined by Larry Peterson, Timothy Roscoe, David Culler, and Thomas Anderson includes support for wide-area network measurement. The assistance of Andy Bavier, Mic Bowman, and Vivek Pai has made my use of PlanetLab much simpler.

I am grateful to the administrators of the traceroute servers whose public service enabled my work and to the operators who provided feedback on the quality of the maps. CAIDA provided Skitter data. Ramesh Govindan helped test the alias resolution technique and provided mapping advice. I also thank Steve Bellovin, Christophe Diot, and Randy Bush for early insights into ISP backbone and POP topologies, and the anonymous ISP operators for comparing the maps I measured to their own. Henrik Hagerstrom assisted in some analyses. Allen Downey provided lognormal distribution analysis tools and guidance. Walter Willinger provided helpful

feedback on the implications of the analysis results. Mark Allman and kc claffy provided helpful support and encouragement. Anukool Lakhina helped apply his sampling biases tests to the Rocketfuel dataset.

Melody Kadenko and Shannon Gilmore ensured that I always had coffee and never had to settle for poorly made espresso beverages (unless I made them myself).

Mike Ernst saved my wrists by introducing me to Kinesis.

Finally, I am grateful to those who bring me their infectious energy and seemingly unstoppable enthusiasm, without whom I would be lost: Vibha Sazawal, Marianne Shaw, and David Wetherall.

Chapter 1

INTRODUCTION

The hardware and software components of the Internet are well-understood. *Links* carry data packets from one router to the next, *routers* receive packets in one link and send them out another, and *routing protocols* choose the paths that packets will take. Although researchers continue to discover unexpected interactions and vulnerabilities in these components, basic information about how they work is well-documented in published standards because such information must be shared for interoperability.

In contrast, how these components are arranged and configured to form the networks in the Internet is not well-understood. The Internet comprises thousands of independently-administered networks. Each network is an *autonomous system* (AS). I focus on a group of autonomous systems called *Internet Service Providers* (ISPs) that provide the service of delivering packets. Distributed administration allows the Internet to grow organically, without centralized design and without the need to reveal internal design choices. Each ISP may have a different business model, and different high-level design goals may influence ISP network design decisions including which cities should be connected, how many links to deploy for redundancy, and how much capacity to provision to support peak load. The independence of different networks means that network operators see and fix problems within their networks, but have little information about how other networks are run.

The subject of this dissertation is how to use the current features of the Internet to discover how its constituent networks are *designed*—how routers and links are deployed and configured in routing protocols. I focus on two fundamental features of Internet design. I first measure *router-level ISP topologies*: how routers are connected in each ISP. Second, I show how to find the *routing policy* of an ISP: how routers are configured to choose which paths to use when forwarding packets. To discover these features of Internet design, I use only the primitives available to *outsiders*: researchers, users, or operators from other ISPs.

Two sources of information make it possible for outsiders to discover the internal structure of ISP networks. The global routing protocol, BGP, allows ISPs to exchange information needed for the Internet to function. This globally-visible routing information consists primarily of the paths of ISPs traversed to reach every destination in the network.[1] ISPs must publish information in this protocol so that neighbors can use the ISP's network to reach destinations. Although others have studied global routing information alone [40, 49, 50, 131], I use it to focus on specific ISPs.

A debugging tool called traceroute provides a second source of information. Traceroute helps network operators diagnose problems and tune the network by showing the sequence of routers used to reach a destination. I use traceroute to discover connections between routers in the topology. Traceroute output often includes meaningful names associated with the addresses in the path. However, traceroute cannot measure arbitrary paths—it can only discover the sequence of router IP addresses visited along paths from the host running traceroute.

I use the information from these routing and debugging interfaces to measure structured router-level ISP topologies and infer network routing policies. The insight that makes measuring ISP topologies practical is to choose only those measurements likely to contribute information about the ISP being measured. To support my analysis of network routing policy, the central insight is that the alternate paths present in the topology but not

[1] Locally-visible routing information within an ISP, on the other hand, may include a complete, router-level topology; this information is hidden from outsiders.

used expose routing policy decisions. The work I describe in this dissertation has provided reasonably accurate, structured, router-level ISP topologies and the first studies of network routing policy at the router-level.

1.1 Benefits of Measured Topologies and Routing Policies

My reason for studying Internet design is one of scientific curiosity: the Internet is a massive, real system built and used by many people. Every piece of the network can be designed independently, evolves over time as the network grows, and can change dramatically as businesses partner or merge. Yet beyond a few standard protocols, little is widely known about how the Internet is constructed and what practices are universal.

An understanding of how the Internet operates in practice is important because it should help researchers recognize and fix the problems the Internet faces [8]. One consequence of the size and importance of the Internet is that it is difficult to change the software it uses: new or updated software that runs on routers in the middle of the network must not only be compatible with previous versions, but must also be well-tested to show its value and that deploying it will cause no interruption of connectivity. These requirements are nearly impossible to fulfill when developing new functionality. Because of this ossification, piecemeal solutions are sometimes preferred [54], leading to the current patchwork of incremental solutions. An understanding of Internet design, and the resulting ability to evaluate solutions in the lab [45], may provide new opportunities to deploy elegant new protocols [133] that would otherwise be considered risky.

As an example, my original motivation for measuring ISP topologies and routing policies was to permit the evaluation of new routing protocols. I had in mind protocols that would enable seamless cooperation between partnered ISPs, that would verify the behavior of competitors, that would repair failures quickly and locally, and that would have primitives expressive enough to permit common policies yet be regular enough to support off-line static analysis. Each of these is an open space for research, in part because the effectiveness of these new protocol features may depend extensively on the real design of the network. For example, realistic patterns of redundancy influence the potential of quickly detouring around failures without complete re-convergence of the routing protocol. As another example, the objectives behind real routing policies can shape the design of alternative routing protocols that are expressive and support off-line analysis. Prior measured and synthetic network maps are inadequate for studying proposals like these because they are incomplete or may not match the relevant properties of the true network maps.

As a second example, measured information about network design may help answer the question of how, or whether, to push security features into the network. Recent research has approached different aspects of network security, such as blocking worm replication [81] and tracing denial-of-service attacks back to the source [93, 111, 113]. Approaches to solving these problems depend, at least in part, on the topology of the network and on how cooperative ISPs are. Specifically, rules that prevent further undesired denial-of-service or worm attack traffic must be propagated back toward the source, crossing ISP boundaries and following multiple paths when necessary. The topology of the network and the routing policies in use can affect how widely improvements must be deployed before they show benefit [93]. More generally, real data can help answer questions about how new features might best be deployed in the Internet, especially those features that require cooperation between ISPs like multicast or quality-of-service.

The ability to discover how networks in the Internet are designed may help with day-to-day operation and debugging. Although information is hidden behind the interfaces that isolate ISPs, problems caused by misconfiguration [74] and failure are not always isolated [19]. For example, the routing protocol messages that carry new or revoked paths when a misconfiguration causes a route to change may propagate across the entire Internet. Further, the protocol lacks features to prevent invalid routes from being introduced. With a greater understanding of how adjacent networks are run, operators may be able to isolate problems more quickly and avoid bad interactions. For example, operators may be able to avoid shifting traffic to an apparently uncongested path that is quite congested downstream in another ISP.

Real data should foster the development of better protocols, systems, and tools, and allow them to be evaluated in a way that provides confidence in the result. Without an understanding of how the Internet is built in practice, some researchers evaluate proposals with synthetic networks or even decades-old topologies [51, 98, 135]. Realism is compromised when using such networks because relevant properties, such as link performance, may not be modeled adequately if at all. With real data, new systems can be developed that exploit common patterns of network design, such as hierarchy and local redundancy. These new systems can then be evaluated in a way that considers the heterogeneity of network designs. Real data about Internet operation may do for network protocol design what file and Web access traces have done for file system and proxy cache architectures: provide the benchmarks by which researchers can evaluate and compare their designs.

In a broader context, understanding the design of ISP networks can be a step toward recognizing what information now hidden *should* be shared by ISPs: what information is relevant for managing, debugging, and tuning the network as a whole [73]. This broader goal is rooted in the belief that the scalability of the network derives not only from what information is hidden, but also from what information is shared. That is, for the Internet to continue to grow, it may be necessary to share more information. In this dissertation, I measure the network using only those primitives available today, and defer determining what information should be shared to future work.

1.2 Challenges and Goals

The following challenges impede measuring structured, router-level ISP topologies and inferring routing policies. Each challenge results from the fundamental and practical features of the Internet. As a result, each challenge implies a goal.

1.2.1 Measuring a Changing Internet at Scale: Efficiency

The Internet is global, densely-connected, and composed of many networks administered by different organizations. The size and density of connections in the network means that many measurements may be needed to discover a complete topology and a complete characterization of routing policy. The distributed administration of the Internet allows it to evolve and grow organically, but continuous change means that measurements must be taken in a short interval: short enough that the network does not change significantly.

This challenge makes *efficiency* in network mapping the primary goal of this work. Unlike prior network mapping efforts that modify measurement tools to achieve efficiency on a small but dedicated platform, I approach the problem by choosing only those measurements likely to contribute data about an ISP.

Efficiency is not an absolute. The specific goal is to have a mapping procedure that is efficient enough that measurements are collected over an interval short enough that the ISP network being measured is relatively stable. “Efficiency” in this context is about being practical, not about avoiding unnecessary work at all costs.

1.2.2 Making the Result Useful: Structure

An important measure of the value of network measurements is whether they provide new insight. The topology of a network, expressed simply as a graph, has been useful for analytical studies [40, 41, 92, 104]. For the network topologies to be useful for simulation and the evaluation of network protocols, I believe they must have *structure*: information to support further understanding, in particular, information about the locations and roles of routers. This information must be inferred from naming information and performance measurements. Prior use of naming information and performance measurements for discovering the locations of routers and hosts in the Internet [47, 85, 91] has been decoupled from forming a topology.

Structure serves two particularly useful purposes in a measured network topology. First, it provides enough context for ISP operators to compare measured maps to their own mental pictures of what their backbone topologies look like. Second, it provides enough information for researchers to start to guess other properties, such as the relationship between geographic distance and link weights in the intra-domain routing protocol.

1.2.3 Narrow Interfaces: Measurement as an Outsider

This dissertation is focused on understanding Internet design as an *outsider* because it is the only way to understand a variety of ISPs. Unfortunately, few, limited primitives are available to outsiders. Outsiders have no access to the information-rich management protocols that can be accessed by those who run networks. The inter-domain routing and debugging interfaces that are available are narrow in that they permit only a limited view of the network: what a particular vantage point can observe at a particular time.

I use primitives that are available to outsiders in the current Internet. I use traceroute to find paths through the network, inter-domain routing information to guide the selection of measurements for efficiency, naming information to annotate routers with locations, and direct probing of router IP addresses to recognize which IP addresses belong to the same router. Some primitives available today may not be supported in the future—many administrators concerned about security disable “optional” services. Conversely, new protocols [36, 71] and systems [116] for Internet measurement may simplify some of the procedures I develop, though the rest of these fundamental challenges, and the solutions I describe to address them, remain.

1.2.4 Tolerating Error and Transients: Accuracy

The measured topologies and routing policies should be *accurate*. One piece of accuracy is *completeness*, finding all the routers and links, which is tied to the challenge of scale above. The second piece is *correctness*, that each found router and link is real and annotated correctly, which is made difficult by the transient events and errors in measurement. As I explain in Section 3.1, traceroute can detect false links, corrupting the measured topology, or traverse paths that are only used during failure, corrupting inferred network routing policy. My procedure for discovering which IP addresses belong to the same router can also err with small probability. Further, my procedure for inferring the geographic location of routers may misinterpret or rely on incorrect naming information. Because many measurements are required to measure a complete network topology, a small fraction of erroneous measurements can create significant inaccuracy.

1.2.5 Verifying the Accuracy of the Result

Although measuring an accurate map in the first place is difficult, demonstrating that the map is indeed accurate is a further challenge. The accuracy of measured topologies and inferred routing policies is difficult to evaluate because the true topologies and router-level configurations are unavailable to me.

My approach to verifying the accuracy of the router-level topologies is to try as many comparisons as possible: to prior measured maps, to routing information, to missing routers found by scanning IP address ranges, to performance information to sanity-check inferred geographic locations, and to the true maps known by a few ISP operators. Each of these techniques is infeasible or inappropriate for network mapping because it does not meet the goals described in this section, but each provides a reference point for reasoning about the accuracy of the map.

My approach to verifying the accuracy of the inferred network routing policies is to determine whether they are predictive. Concretely, that the inferred (“learned”) rules can be trained on a subset of observed paths and then predict how the remaining paths are chosen. Accuracy in recovering the exact routing protocol configuration parameters these rules represent is unattainable: (almost) infinitely many possible configuration parameter choices would result in the same behavior. That the routing protocol model is predictive makes it accurate enough to speculate about the higher-level goals of the routing policies.

1.2.6 Summary

The fundamental challenges described in this section yield the following four goals: *accuracy* in the measured topologies and inferred policies, *efficiency* in measurement, *structure* in the measured topologies, and using only primitives available to an *outsider*. In this dissertation, I will show that each goal is achieved.

1.3 Thesis and Contributions

In this dissertation, I support the following thesis: *outsiders can use routing and debugging information to efficiently measure accurate and structured, router-level ISP topologies and infer ISP routing policies*. To support this thesis, I develop, apply, and evaluate a set of techniques for efficiently and accurately measuring structured topologies and inferring routing policies. I use these techniques to discover previously unknown structure and policy characteristics of the Internet.

This dissertation makes the following contributions:

Techniques for mapping ISP network topologies. The techniques I develop for mapping ISP network topologies are based in the philosophy of Internet measurement affirmed in this dissertation: aggressively avoid unnecessary measurement and apply this efficiency to gain accuracy.

I present and evaluate three primary techniques. The first technique applies global routing and prior measurement information to select traceroute measurements likely to contribute information about an ISP network. These techniques chose fewer than one thousandth of the measurements that would have been taken by a conventional approach that measures from all vantage points to all destinations. The second technique directly probes pairs of IP addresses to find those that belong to the same router, which is a new approach to the problem of *alias resolution*. Again, to be efficient, only a small, selected subset of all-pairs of IP addresses may be tested. This alias resolution technique can find twice as many IP address aliases as prior techniques. The third technique uncovers location and role information embedded in DNS names to annotate the network topology. Most recovered geographic locations are consistent with observed measures of performance, and geography provide structure to the measured topologies that makes them more useful for understanding the design of ISP networks.

Techniques for inferring intra-domain and peering routing policies. The techniques I develop for inferring routing policies are based on the insight that reasonably accurate network maps include many paths *not* taken, and that these alternate paths expose the choices of routing policy. These techniques provide the first look at intra-domain and peering routing policies across several ISPs.

I present and evaluate techniques that infer both intra-domain and peering routing policies. The inference of intra-domain routing policy is based on a constraint system in which the cost of observed paths is less than the cost of alternate paths. Solving the constraint system, while tolerating incomplete and erroneous measurements, yields a set of link weights consistent with routing. These link weights correspond to the configuration of the network, but are not the true link weights used. The inference of peering routing policy is based on a classification that determines whether the observations are consistent with common policies of early- and late-exit or consistent with some alternate, high-level approach.

An analysis of the characteristics of measured ISP network topologies and routing policies. I measure two large datasets, one of the router-level topologies of ten ISPs collected simultaneously, and a second of the POP-level topologies of 65 ISPs. The router-level topologies were collected to a level of accuracy roughly seven times as complete as previous efforts. The POP-level topologies, which are coarser in that each node is a city, were measured to provide input to the routing policy analysis.

I use the policy-annotated, structured ISP network topologies to conduct the first quantitative study of the similarities and differences in how ISPs engineer their networks, finding significant heterogeneity in backbone designs. I also find that highly-skewed distributions pervade attributes of these topologies: most POPs are small but most routers are in relatively large POPs; most routers connect to only a few neighbors, but a few connect to hundreds; most pairs of ISPs connect in only a few places, but a few connect in tens of cities.

This study of network routing policy in the Internet is the first of its kind and has provided new insight. Intra-domain routing policy is consistent with shortest-latency routing and thus consistent with adequate provisioning of the network. Peering routing policies are diverse, even to different peers of the same ISP. Peering routing policies are often asymmetric: in particular, late-exit is unlikely to be reciprocated.

1.4 Organization

In Chapter 2, I provide some background for understanding the contributions made in this dissertation, with a focus on the terminology of Internet topology and routing. In Chapter 3, I describe related work organized by the basic information sources used in Internet measurement: traceroute debugging, inter-domain routing, and DNS naming information. In Chapter 4, I detail my approach to ISP topology measurement, including techniques for focusing on individual ISPs, resolving IP aliases, and recovering the structure of the topology. The focus of Chapter 5 is evaluating the accuracy and efficiency of the mapping techniques. In Chapter 6, I present and evaluate my approach to inferring predictive models of network configuration in the form of approximate routing policy. In Chapter 7, I present the measured maps and analyze both topologies and routing policies in some detail. I conclude in Chapter 8 with lessons for future efforts and a description of future work in understanding Internet design.

Chapter 2

BACKGROUND

This chapter provides top-down overviews of Internet topology and routing protocol concepts. Background material on the network measurement methods used by related work appears in the next chapter.

The reader should already be familiar with basic networking terminology, including terms such as packet, router, link, and address, that would be found in the first chapter of an introductory textbook. This chapter introduces concepts and terminology regarding Internet operation, and may be skipped or skimmed by readers familiar with terms including localpref, MPLS, and early-exit routing.

I use italics for specific terms that are used later in this dissertation and quotation marks for informal terminology.

2.1 Internet Topology Concepts

A network topology is a graph of nodes and edges. The Internet has several different types of network topology, and for the most part, these topologies can be organized by level of granularity. In this section, I define three levels of Internet topology starting with the highest: inter-domain,[1] POP-level, and router-level. The "nodes" of the inter-domain and POP-level topologies represent large aggregates. In later chapters, I will measure POP- and router-level topologies.

I now describe each of the three levels of Internet topology.

2.1.1 Inter-domain Topology

Figure 2.1 is an illustration of an inter-domain topology. Each node in the inter-domain topology is an AS, represented by a cloud in the figure. Those ASes that provide service to others are ISPs. Each edge in this topology represents a business relationship that results in the exchange of Internet traffic between ASes. Most edges connecting large and small ASes represent provider-customer relationships in which the smaller AS pays the larger for *transit service*, the service of providing connectivity to the rest of the Internet. Comparably-sized ASes may negotiate the exchange of traffic without exchanging money as a way to avoid sending traffic through a provider. This creates an edge between "peers."

Not all ASes in the topology are alike. A clique of *tier-1 ISP* networks, which are the largest network providers, compose the "core" of the inter-domain topology. Many tier-1 ISP networks span continents. (A complete list of the tier-1 ISPs in early 2003 appears in Table 6.1 on page 75.) Tier-2 ISPs, many of which are regional network providers, connect to some tier-1 ISPs. Most medium-sized ISP networks and large customer networks are *multi-homed*: they connect to more than one provider. Multi-homing improves reliability and performance [1, 112, 123]. (An AS may also be multi-homed within a single provider if it connects to it in two different places; this level of multi-homing does not appear in the inter-domain topology.) Furthest from the core are the *stub* networks that are "leaves" in the topology, connecting to only one provider. The smallest networks may be multi-homed or stubs, but provide service to no other network. Although each stub network participates in inter-domain routing, the decisions each makes are of little consequence when only one path connects it to the rest of the Internet. This classification of ASes is informal and not precisely defined, but is illustrative of common relationships between networks in the Internet.

[1] I use "inter-domain topology" instead of "ISP-level topology" to avoid confusion with "ISP topologies."

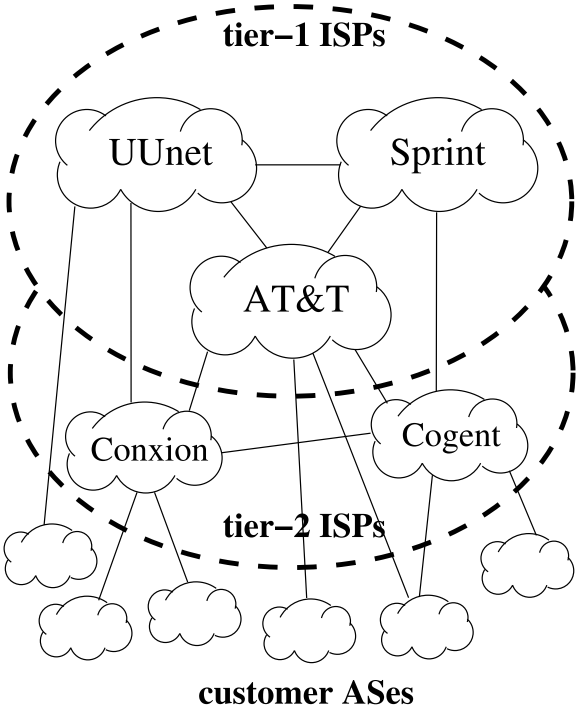


Figure 2.1: An illustration of an inter-domain topology. The largest, tier-1 ISPs are connected in a clique. Smaller, tier-2 ISPs connect to some of the tier-1 ISPs and possibly to each other. Customer ASes at the bottom may connect to multiple providers. Large customers may connect directly to tier-1 ISPs.

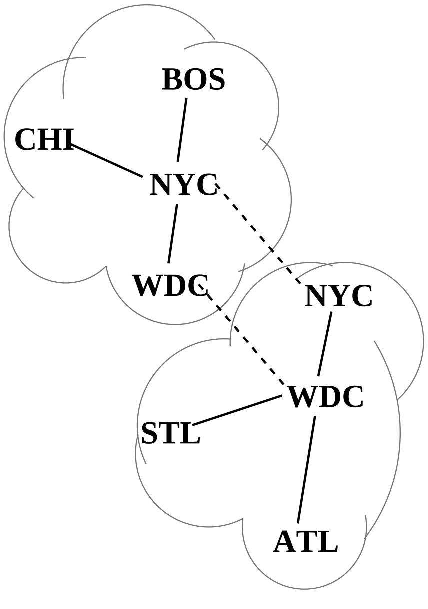


Figure 2.2: An illustration of a POP-level topology including two ISPs are shown. At left is an ISP with Boston, New York, Washington D.C., and Chicago; at right one with New York, D.C., St. Louis, and Atlanta. Solid lines connect POPs within an ISP and dotted lines represent peering links between POPs in different ISPs. Most peering links stay within the same city.

2.1.2 POP-level Topology

Adding information about geographic location to the inter-domain topology produces the POP-level topology. Each router in the Internet is housed in a building that provides cooling, power, and proximity to other routers. A *point of presence (POP)* is the collection of routers owned by an ISP in a specific geographic location (city or suburb).[2] Different ISPs are likely to have routers in the same building, such buildings are called exchange points or co-location facilities; I treat a POP as a logical entity specific to an ISP. The topology of the routers within a POP is likely to differ significantly from the topology of the whole network: links within a POP are likely to be much less expensive and thus plentiful, and this smaller topology should be simpler to manage.

Figure 2.2 is an illustration of a POP-level topology. Each node is a POP, identified by both ISP (the cloud) and city (the three-letter abbreviation). Edges between POPs may represent backbone links within an ISP or peering links to other ISPs. *Peering links*, the connections between ISPs, may be at *exchange points* where several ISPs have routers in the same building, or may be *private peerings* where two ISPs perhaps lease a connection between different buildings. Private peering links may connect different buildings in different suburbs, but rarely do they connect different metropolitan areas.

The POP-level topology is useful for understanding the geographic properties of Internet paths. It provides straightforward performance constraints: the latency between POPs in two cities is at least the time it takes for light to traverse the path.

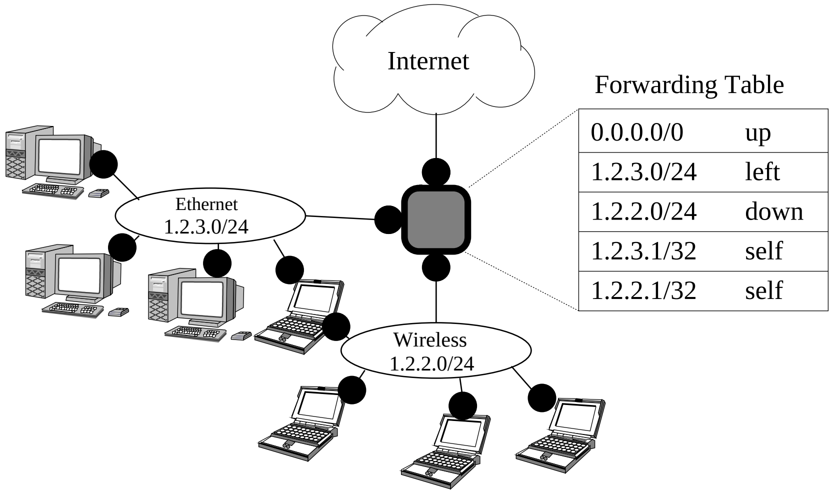
2.1.3 Router-level Topology

In a router-level topology, each node is a router and each link represents IP-level connectivity between routers. Each *router* is like a desktop computer in that it has a processor and an operating system. Unlike a desktop computer, a router has many network interfaces to connect to other routers and has been designed to *forward* packets: to receive packets on one interface and send them out another. The "routers" consumers can purchase at an electronics store are small scale versions of the routers operated by an ISP: they have multiple interfaces, possibly of different types (many home routers have both wired and wireless interfaces), they have a processor with software, and their primary purpose is to forward packets from one link to the next. The *links* in the router-level topology represent IP-level connectivity: a link connects two routers if the other router can be reached in one IP-layer (network-layer) hop. These point-to-point links may not be point-to-point beneath IP: a layer-2 switch or other multiple-access medium may be used.

To describe how IP addresses are allocated to network interfaces, I will use the small office network topology illustrated in Figure 2.3. The shaded box in the middle represents a router that has three interfaces represented by solid circles. Below the router is a wireless network used by laptops. Above the router is the rest of the Internet. To the left of the router is a wired Ethernet network used by workstations. The Ethernet segment has a subnet *prefix*: 1.2.3.0/24. This notation means that the prefix is 24 bits long: that the first 24 bits of the 32-bit IP address are fixed, which leaves room for 8 bits of addressing for hosts on that subnet. A /24 prefix is a convenient unit of allocation for subnets because it allows 253 addresses for machines (three of the 256 are special or used by the router) and the prefix address (1.2.3) is visually separated from the host address in the last octet. A *larger prefix*, such as a /16, has more addresses and a shorter prefix length; a *smaller prefix*, such as a /28, has fewer addresses and a longer prefix length.

Every router maintains a *forwarding table*, which, in this example, needs five entries. A forwarding table maps destination prefixes to outgoing interfaces where packets should be sent. When looking up a destination in the forwarding table, more than one prefix may match; the entry for the longest prefix is used. At the top is a *default route*: the entry to use when no other entry is matched. The default route is associated with the prefix 0.0.0.0/0, which is the least-specific prefix possible: any other matching prefix will override this default

[2] More specific terms include *data-center* or *exchange*; for this dissertation, I use *POP* to mean a distinct location that houses any number of ISP routers for any purpose.



Forwarding Table

0.0.0.0/0	up
1.2.3.0/24	left
1.2.2.0/24	down
1.2.3.1/32	self
1.2.2.1/32	self

Figure 2.3: IP address prefixes are assigned to network links. Host interfaces have IP addresses that are members of a network link's IP address prefix. The forwarding table at each router needs only an entry for each subnet, not for each host.

route by virtue of being more specific. Each packet not matched by a more-specific prefix will be sent out the "up" interface, toward the rest of the Internet. Next are entries for both of the adjacent subnets. Each of the workstations and laptops will have addresses within these prefixes: the router need not know about individual hosts. Last in this table are the two local delivery routes. The router, like each of the hosts, has an interface on each network to which it attaches. By convention, the interface on the router that provides service to the rest of a subnet takes the first ".1" address in the subnet. The router, thus, accepts traffic to 1.2.3.1, which allows the router to communicate for administration and monitoring.

Figure 2.3 shows two important features. First, hierarchical addressing allows forwarding tables to remain small and limits the size of the routing protocol traffic exchanged. Small forwarding tables fit in smaller, faster memory, allowing higher performance forwarding. The rest of the Internet needs only one entry for this network: 1.2.2.0/23. That only this aggregate is exposed to the rest of the network will help limit the number of unique destinations that a network mapping project must measure: all of the addresses in this prefix will be routed through the rest of the Internet along the same path. Second, IP addresses belong to interfaces; a machine with more than one interface will have more than one IP address. Although a router is a perfect example of a machine having many interfaces, laptop computers may also attach to a wired and wireless network at the same time. Such machines have multiple interfaces, and thus multiple addresses, but usually do not act as routers: they do not forward traffic from one network link to another.

The networks run by ISPs are IP networks like this network of laptops and workstations, but differ in several ways, some of which I list here. First, the routers have many more interfaces and the subnets are much smaller because they connect relatively few routers. Second, a routing protocol configures the forwarding table: this smaller network is simple enough to be configured by hand. Third, default routes are not used in the core of the Internet; routing is said to be "default-free" in the middle of the Internet [48].

Each router also has a *role* in the topology. Routers with links that connect to other POPs are *backbone routers*, which typically have relatively few, high capacity links. Routers that aggregate many low-capacity connections to customers into higher-capacity links to the backbone routers are *access routers*. (Access routers may also be known as gateway routers.) Access routers typically do not connect to access routers in other POPs.

In summary, IP addresses refer to interfaces; each IP address is a member of a prefix that represents a network link; routers forward packets from one link to another; each router uses a forwarding table to determine on which link a packet should be forwarded; each entry in a forwarding table corresponds to a prefix; and the most specific (longest prefix) match found in a forwarding table is used.

2.2 Internet Routing Policy Concepts

The configurations of routing protocols determine how packets traverse each of these levels of Internet topology. A *routing protocol* is responsible for exchanging information about the state of the network and deciding which paths to use to reach every destination. The output of the routing protocol is a forwarding table as described above.

The primary role of a routing protocol is to detect and avoid failed links, but it also allows operators to express preferences for different paths to shape how traffic flows across a topology. Some paths may be preferred because they are lightly-used, cheaper, or more reliable. The preferences for different paths constitute *routing policy*, which ISP operators express in the configuration of routing protocols.

In this section, I present an overview of BGP operation and intra-domain routing.

2.2.1 Inter-domain Routing with BGP

Each ISP runs the inter-domain routing protocol, *BGP* [106], to choose routes across the inter-domain topology. BGP is an acronym for *Border Gateway Protocol*, and is a type of distance-vector routing protocol. A *distance vector routing protocol* is a protocol in which each router tells its neighbors the length of the best path to reach each destination. Each router in a distance vector protocol stores a *routing table* that has all the information of a forwarding table (Figure 2.3) and also includes the length of each path. Every router periodically sends a copy of this routing table to each of its neighbors. When a router receives a copy of its neighbor's table, it updates its own, deleting routes that are no longer available from that neighbor and choosing routes that traverse shorter paths.

BGP implements a variant of the distance vector approach; it is a *path-vector routing protocol*. Rather than exchange and increment the length of the path, in a *path-vector* protocol, routers exchange whole paths and each router adds itself to the path as it propagates the route. The path length remains the default metric, though it is no longer passed explicitly but calculated from the path itself. The complete path of ISPs helps avoid routing loops. A *routing loop* occurs when routers choose the next hop toward a destination in a way that the resulting path includes a cycle—a packet will loop around the cycle and not reach its destination. Routing loops are not intentional: they occur when different nodes have inconsistent information about the state of the network. By incorporating a path of ISPs into the routing protocol, each ISP can verify that it is not already part of the path before incorporating the route, avoiding some routing loops.

Figure 2.4 shows a simplified sample BGP routing table. Three paths reach 1.2.3.0/24. These paths are expressed as lists of Autonomous System Numbers (ASNs). Each ASN represents an AS in BGP. Of the three paths to 1.2.3.0/24, the path “11 7 5” has the shortest *AS-path length*: the number of ASes traversed to reach the destination. Packets are forwarded along the path from AS 11 to AS 7 to AS 5. Because route advertisements propagate in a direction opposite to data flow, when a router propagates a BGP routing update, it prepends its ASN to the front of the path. Because each path to 1.2.3.0/24 starts at the right with AS 5, that AS *originates*, or “owns,” the prefix. BGP implementations store a configurable number of alternate paths so that if the “best” path is withdrawn, an alternate can be used quickly.

BGP has limited support for traffic engineering [103]. For paths that are advertised to others, an ISP has two main ways to influence the routing decisions made by other ISPs. First, an ISP may insert its own ASN into a path more than once, a trick known as *AS path prepending*. Although at first this practice may appear to indicate a routing loop, it allows ISPs to make a path available for robustness but discourage its use. AS path

```
1.2.3.0/24          13 4 2 5
                    6 9 10 5
                    11 7 5

4.5.0.0/16          3 7 8
                    7 8
```

Figure 2.4: A sample BGP table snippet. Destination prefixes are on the left, AS-paths on the right. ASes closer to the destination are to the right of the path. AS 5 “originates” the prefix 1.2.3.0/24, and AS 8 “originates” 4.5.0.0/16.

prepending is a coarse tool, however, because it may break many ties at once. Second, IP address prefixes may be disaggregated for finer control, and the routing protocol is obliged to choose the path for the most-specific prefix, as described above in Section 2.1.3. Selection of the most-specific prefix is common for all routing protocols, but can be exploited in BGP to balance load across incoming links on a finer granularity than possible with AS path prepending alone.

To choose between paths that are accepted from different neighbors, BGP uses a parameter called *localpref* (“local preference”). Local preference allows an ISP to rank routes in any order. Common practice is to first choose routes from customers, then choose routes from “peers.” Peers, in this context, are the comparably sized ISPs that exchange traffic with little exchange of money.[3] If neither a customer or peer has a route, choose a provider. This order is a result of the business of Internet service: delivering packets through customers creates revenue, while delivering packets through providers costs money. Delivering packets through peers commonly costs nothing. ISPs with different providers might use localpref to choose a monetarily cheaper or higher performance route. Local preference is associated with each individual route, but may be configured in groups of routes (such as all routes from a customer) or specific routes (the route to 1.2.3.0/24 from one neighbor) [43]. Local preference is thus a low-level parameter often used to implement high-level routing policy goals.

2.2.2 Peering Routing with BGP

ISPs can use routing policy to choose the router-level path that will be used to reach a neighboring ISP. The main decision is, for packets that must cross both ISP networks, which ISP will do most of the work? Figure 2.5 shows two straightforward possibilities. The first is *early-exit*: choose the closest peering point. Early-exit routing can be thought of as a greedy algorithm; it is also the default. The second straightforward policy is *late-exit*: choose the peering point closest to the destination. Although BGP hides information about internal topology, a downstream ISP can export *MEDs* (Multi-Exit Discriminator attributes) that provide a preference order for each of the different peering points that might be used. The highest preference expressed by the downstream is typically for the late-exit. The late-exit of Figure 2.5 could be implemented by advertising routes to prefixes in ATL with a MED of 20 from WDC and a MED of 10 from NYC. MED values do not compose with link weights in intra-domain routing, so it is difficult to find globally-shortest paths. Because exporting MEDs from one ISP and accepting MEDs in another are both optional behaviors, enabling late-exit requires cooperation from both ISPs. Further, late-exit is unlikely to be much better than early: the late-exit path is symmetric with the early-exit chosen in the opposite direction.

[3] *Peers* may also refer to any neighboring network, though usually adjacent networks that are not of comparable size are called *BGP peers* to avoid confusion.

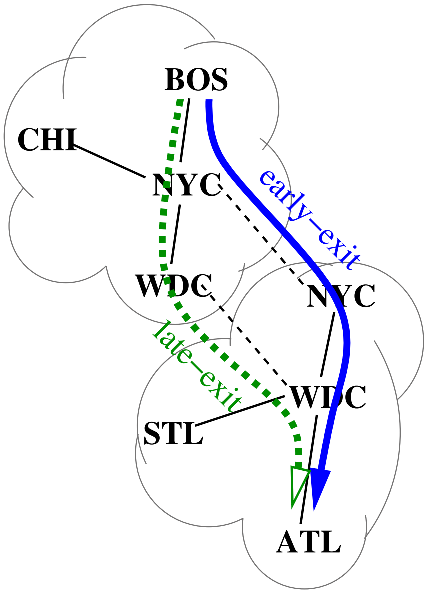


Figure 2.5: Early- and late-exit routes from Boston in one ISP to Atlanta in the other, overlaid on the topology of Figure 2.2. The early-exit is New York, following the solid path, and the late-exit is in D.C., following the dotted path.

Early-exit routing is one cause of *asymmetric routing*: the path used to reach a destination and the path in the opposite direction may differ. Late-exit routing and BGP settings that choose different AS paths in different directions can also cause asymmetry. The significance of asymmetric routing for understanding the Internet is that only the outbound paths are visible.

2.2.3 Intra-domain Routing

Each ISP, in addition to running BGP, runs an *Interior Gateway Protocol (IGP)* to choose paths within the ISP's network. Typical IGPs, like OSPF and IS-IS, use link-state routing instead of the distance vector approach used by BGP. *Link-state routing* is characterized by the exchange of fragments of the topology so that each router can assemble the fragments into a global picture and choose paths from a complete topology. I leave the details of how link-state routing protocols assemble a globally-consistent picture of the topology to the protocol specifications [82, 90]; relevant for this dissertation is how routing policy is expressed by network operators. To set routing policy, a network operator configures a *weight* (or *cost*) for every link. The routing protocol chooses paths that have the least cost: paths having the smallest sum of the weights of the links. The cost of any link can be increased to discourage traffic on, for example, congested or unreliable links. As new information is sent to the rest of the network, different routers may have inconsistent information about the state of the network, which may cause transient behavior.

Some ISPs explicitly configure the path between each pair of routers using MPLS [110]. MPLS is an acronym for multi-protocol label switching. When a packet enters the network of an ISP using MPLS, a router assigns a label to the packet; later routers make forwarding decisions based on this label, not on the destination address of the packet. Because the label is assigned at the first router, forwarding decisions at each hop can depend on the source and destination as well as other fields in the packet. MPLS can cause further trouble for network mapping, a problem which I will discuss in Section 3.1 after I have described how network mapping studies work.

Chapter 3

RELATED WORK

In this chapter, I describe the three main information sources I use in this dissertation: traceroute measurements, global routing tables, and router address names. I present related work organized by the information sources used.

3.1 Traceroute and Active Measurement

Traceroute is a simple, popular tool for discovering the path packets take through a network. It was first written by Van Jacobson at the Lawrence Berkeley National Laboratory [64], but has since been extended with various features [52, 76, 128] and modified to measure more properties [38, 63, 72]. In this section, I first describe how traceroute works, then present related work that uses traceroute for network mapping, and finally summarize its limitations.

3.1.1 How Traceroute Works

Because of the possibility of transient routing loops, every IP packet sent into the network includes a *time-to-live (TTL)* field. At every router, the TTL field is decremented, and if it ever reaches zero, the router sends a "time-exceeded" error message back to the source. If a packet enters a routing loop, the TTL ensures that the packet will not consume unbounded resources: it may loop a few times, but will eventually expire.[1]

Traceroute sends packets into the network with artificially small TTL to discover the sequence of routers along a path. I show sample traceroute output in Figure 3.1. The first packet it sends has a TTL of 1; this packet discovers an address of the first router along the path. Traceroute increases the TTL until it receives a different error, "port-unreachable," which signifies that the packet reached the destination. Traceroute will also stop if it reaches a maximum TTL. (The maximum possible TTL is 255, but most traceroute implementations stop at 64 or 30 because most paths are not so long.)

Traceroute is fundamentally a network debugging tool. It can show the first part of a path up to a failure. By showing which path was chosen, it allows operators to test the configuration of routing policy in BGP and the IGP. Traceroute is also inherently asymmetric: it only discovers the path used to reach a destination; it cannot discover the return path. As a debugging tool, it cannot differentiate problems on the outbound path from problems on the inbound: the loss of either the TTL-limited probe or the time-exceeded error message both prevent a response packet from returning to the source.

Because of the asymmetry of network routing, the utility of traceroute for network debugging, and the ease with which Web services can be provided, public traceroute servers have emerged as a widely deployed platform for network debugging. A *public traceroute server* is a machine, typically a Web server, that will execute traceroute to any given address on request. Operators tuning inter-domain routing can use a public traceroute server to verify that the correct paths are being chosen. Hundreds of public traceroute servers form a loosely-organized debugging facility for the Internet. Unlike the dedicated measurement infrastructures that can run hundreds of traceroutes in parallel for the studies below, public traceroute servers do not support heavy use.

[1] Even the time-exceeded error message includes a TTL in case the return path also has a loop. Error messages are not sent when error messages cannot be delivered, so the resources consumed by a looping packet are limited.

```
electrolite:˜> traceroute www.cs.umd.edu
traceroute to www.cs.umd.edu (128.8.128.160), 64 hops max, 40 byte packets
 1  eureka-GE1-7.cac.washington.edu (128.208.6.100)  0 ms  0 ms  0 ms
 2  uwbr1-GE2-0.cac.washington.edu (140.142.153.23)  0 ms  0 ms  0 ms
 3  hnsp2-wes-ge-1-0-1-0.pnw-gigapop.net (198.107.151.12)  7 ms  0 ms  0 ms
 4  abilene-pnw.pnw-gigapop.net (198.107.144.2)  0 ms  0 ms  1 ms
 5  dnvrng-sttlng.abilene.ucaid.edu (198.32.8.50)  34 ms  26 ms  26 ms
 6  kscyng-dnvrng.abilene.ucaid.edu (198.32.8.14)  37 ms  37 ms  37 ms
 7  iplsng-kscyng.abilene.ucaid.edu (198.32.8.80)  249 ms  235 ms  207 ms
 8  chinng-iplsng.abilene.ucaid.edu (198.32.8.76)  50 ms  50 ms  58 ms
 9  nycmng-chinng.abilene.ucaid.edu (198.32.8.83)  76 ms  73 ms  76 ms
10  washng-nycmng.abilene.ucaid.edu (198.32.8.85)  74 ms  75 ms  74 ms
11  dcne-abilene-oc48.maxgigapop.net (206.196.177.1)  74 ms  74 ms  74 ms
12  clpk-so3-1-0.maxgigapop.net (206.196.178.46)  75 ms  75 ms  75 ms
13  umd-i2-rtr.maxgigapop.net (206.196.177.126)  75 ms  75 ms  75 ms
14  Gi3-5.ptx-fw-r1.net.umd.edu (129.2.0.233)  75 ms  75 ms  75 ms
15  Gi5-8.css-core-r1.net.umd.edu (128.8.0.85)  75 ms  75 ms  75 ms
16  Po1.css-priv-r1.net.umd.edu (128.8.0.14)  75 ms  75 ms  75 ms
17  128.8.6.139 (128.8.6.139)  75 ms  75 ms  75 ms
18  www.cs.umd.edu (128.8.128.160)  75 ms  75 ms  75 ms
```

Figure 3.1: Traceroute output from the University of Washington to the University of Maryland. Each line presents the result of sending three probes with the same TTL. This result includes the address and DNS name of the source of the responses and the time to receive each of three responses.

3.1.2 Related Traceroute-based Mapping Studies

In this section, I describe recent Internet mapping approaches. These techniques are relevant because they address the challenges necessary to build a global picture of the network. In this section, I contrast the work of Pansiot and Grad [92], Govindan and Tangmunarunkit (Mercator) [55], Burch and Cheswick (Lumeta) [25], and claffy, Monk and McRobb (Skitter) [33]. Renderings of Mercator and Lumeta maps appear in Figure 3.2.

Prior mapping efforts can be classified by how they choose destinations, how many sources are used, whether they provide an IP- or router-level map, and whether they provide a snapshot or a persistently-updated view of the network. Some also develop techniques for efficient network mapping and alias resolution.

Most mapping projects choose relatively few destinations. Finding and managing a set of destinations is a somewhat difficult technical problem. A good destination is *responsive*: when it receives a probe packet, it will send a response. This means that the destination should be a machine that is usually on. Administrators of some hosts in the network object to receiving unsolicited packets; these destinations must be removed.

Pansiot and Grad [92] measured a network topology using traceroute so that they could evaluate multicast protocol proposals that might use less state in some routers. Pansiot and Grad collected two data sets. First, twelve sources ran traceroute to 1,270 hosts, and one source (their own) ran traceroute to 5,000 hosts that had previously communicated with their department. To run their measurements efficiently, Pansiot and Grad modified traceroute in two ways. First, they did not probe three times per hop, but instead returned after the first successful response (retrying if no response was received). This can reduce the packet cost of traceroute-based mapping by two-thirds. Second, they configured traceroute to start probing some number of hops into the trace, avoiding repeated, redundant probes to nearby routers. They also pioneered work on alias resolution, introducing a test that detects an alias when two different addresses respond to probe packets using a common source address. This test needs only one probe packet per discovered address.

Burch, Cheswick, and Branigan [25, 30] explore a different point in the design space: many more destinations (ninety thousand) but only a single source host. This work is primarily concerned with understanding small components, tracking change over time, and visualizing the overall network, and has resulted in Lumeta,

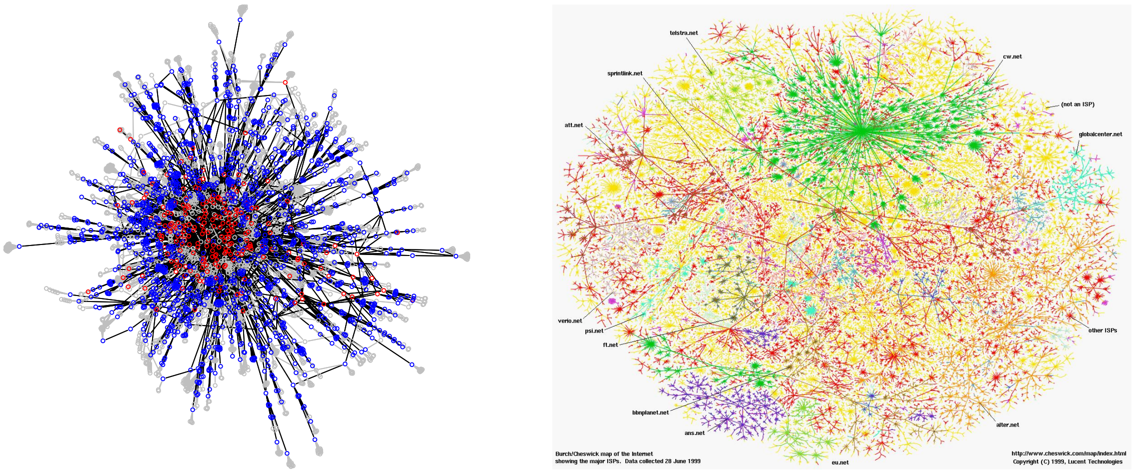

telstra.net
sprintlink.net
att.net
cw.net
(not an ISP)
globalcenter.net
verio.net
psi.net
ft.net
other ISPs
bbnplanet.net
ans.net
alter.net
eu.net
Burch/Cheswick map of the Internet
showing the major ISPs. Data collected 28 June 1999
http://www.cheswick.com/map/index.html
Copyright (C) 1999, Lucent Technologies

Figure 3.2: Internet map visualizations from different projects. At left is Mercator's view of an ISP named Cable and Wireless. At right is Burch and Cheswick's map of the Internet from 1999; Cable and Wireless is the green star in the upper right labeled `cw.net`. A larger picture is available on-line [24]. These maps show two different structures for the same network.

a company specializing in mapping networks. Burch *et al.* explore a different point in the design space: many more destinations, but just one source.

Govindan and Tangmunarunkit [55] in the Mercator project added "informed random address probing," in which they chose traceroute destinations by consulting the global Internet routing table. They showed that routing information could parameterize network mapping. Govindan, like Burch, uses a single source, but use source routing to "bounce" probes off remote routers to create many virtual sources. Govindan and Tangmunarunkit extend the basic alias resolution technique of Pansiot and Grad in a similar way: they use many virtual sources because some addresses cannot be reached from every source.

CAIDA's Skitter [22, 33] project increases the number of destinations by picking Web servers (originally 29,000, though this has increased over time). They use six DNS servers as *vantage points*: places in the network that run traceroute. (This platform has since grown to over 26 servers). Like Burch, the Skitter project maintains a history of previous measurements to support studying change in the network.

These Internet mapping techniques focus specifically on the graph of connectivity between routers. They do not provide structured topologies with POPs. The whole Internet is a consistent goal, feasible only by sampling the topology—choosing limited destinations (Pansiot and Grad, Skitter), or choosing a single source (Lumeta)—or by running for weeks (Mercator).

3.1.3 Challenges and Limitations

The limitations of using traceroute for Internet mapping are several; these limitations affect how the maps should be used and the possible accuracy with which they can be measured.

Traceroute cannot discover links that are not used. Unused links include backup links that become active only when another fails. This limitation means that traceroute-based Internet maps should not be used in quantifying the amount of redundancy in the network. Restated, traceroute-based maps cannot show that a network is vulnerable to the loss of a link—an unseen backup link may exist. The severity of this limitation is an open question: how many links are created but inactive until a failure occurs?

Traceroute cannot discover links that are not used on paths from the measurement vantage points. With only one vantage point, the network map would look much like a tree. A single vantage point would not discover

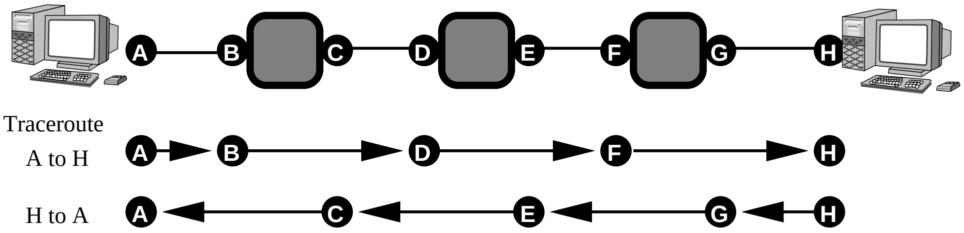


Figure 3.3: Because traceroute lists input interfaces, it is not obvious when traceroute observes the same path in opposite directions.

cross-links in the topology. As a result, traceroute-based mapping studies see dense fanout, or “bushiness” close to measurement sources and long chains further away [69]. I use more, diverse vantage points because the larger measurement platform is likely to yield a more complete map, but the severity of this limitation remains an open question: how many vantage points are needed?

Traceroute may show false links due to routing changes during a measurement. If a routing change modifies the length of a path at the same time that traceroute increments the TTL it sends in probe packets, a false link segment may appear. A challenge for network mapping is to remove such false links. I remove links that have egregiously high link weights (routing policy appears not to use them) or are seen too few times for their apparent importance (a direct link from Seattle to London would be attractive, but if rarely used, it is probably not real). An alternate approach would be to redesign the traceroute tool to be less vulnerable to routing changes; unfortunately, deploying such a modified tool widely would require a great deal of effort.

Traceroute discovers IP-level connectivity, not the physical and data-link topologies underneath. For example, routers on a single switched network may appear to traceroute as a clique of independently connected routers. Patterns of IP address allocation like those shown in Figure 2.3 may expose which links are likely to be shared and which are likely to be point to point [115].

The IP addresses discovered by traceroute represent the interfaces that received traceroute probes. As a result, traceroutes traversing the same path in opposite directions may not see an IP address in common, as shown in Figure 3.3. That mapping by traceroute requires a separate alias resolution phase to determine which addresses represent the same router is a challenge I address in Section 4.2. The IP addresses discovered may be unsuitable for the alias resolution techniques because they use private address space [107], unroutable address space, are unresponsive to further probing, or, worse, are anonymous in that they do not respond even to initial traceroute probes [134]. These problems are rare in my experience with the ISPs I studied, but can prevent an accurate topology from being assembled.

Some MPLS networks hide the underlying topology by disabling the TTL used by traceroute. Routers using MPLS may be configured either to decrement the TTL, as traceroute requires, or to ignore the TTL field: because the switched paths of MPLS are configured to have no loops, the IP TTL is not needed. The MPLS specification, however, recommends that the TTL be decremented where possible [110, Section 3.32].

Finally, traceroute requires limited cooperation from ISPs—the packets used by traceroute may be blocked. This means traceroute cannot be used to map large enterprise networks that are behind firewalls. It also means that ISPs have a means to avoid having their networks mapped by traceroute.

3.2 BGP Routing Information

Views of global BGP routing information make it possible to collect information about the network topology without sending measurement packets into the network. In this section, I describe the Route Views project that collects and archives BGP routing information from many ISPs, how this information has been used in research, and the limitations of the partial views of global routing that Route Views offers.

3.2.1 The Route Views Archive

David Meyer's Route Views [80] project from the University of Oregon publishes a continually updated archive of BGP updates from 53 different ISPs in the Internet.[2] It uses these updates to maintain a large view of inter-domain routing. This view is still partial, both because not all ISPs are represented and because different routers within the same large ISP may choose different paths.

Because BGP is a path vector routing protocol, each advertised route includes a list of autonomous systems used to reach a destination prefix. When paths used to reach all IP address prefixes from several ISPs are combined in an archive like Route Views, they provide a rich source of data about inter-domain network topology and routing policy. For example, a partial view of the inter-domain topology can be constructed and the paths chosen along this topology can be used to infer provider-customer relationships.

The usefulness of BGP information has been widely recognized due to this and other work [75]. As a result, BGP information is provided directly to many end-hosts that participate in network measurement, such as nodes in PlanetLab [84, 97]. The different perspective edge networks have is likely to complement the BGP information stored in Route Views. I focus on Route Views as the largest source of BGP information today. Although BGP information may be more widely distributed in the future, it will likely have similar limitations.

3.2.2 Prior Work Using Route Views

Work that uses Route Views data include Gao's inference of inter-domain policy [49] and Faloutsos's discovery of power-laws in the inter-domain topology [40]. Many other studies have used Route Views data; the two I highlight are relevant for the later analyses of router-level topologies and routing policies in this dissertation.

Michalis Faloutsos *et al.* [40] analyzed the inter-domain topology that can be inferred from Route Views BGP and found that it could be characterized by various *power-laws*. A power law is an expression of the form $y \propto x^a$ where a is a constant, and x and y are the related terms of interest. Faloutsos found power laws in a few graph properties in the inter-domain topology. Foremost is the degree distribution: the distribution of the number of neighbors of each node. Informally, a few ASes have very high degree and most have very low degree. It is likely that the nodes with many neighbors are more important in maintaining connectivity. Another property is the neighborhood size: how many nodes can be reached in a fixed number of hops. They showed that most of the network could be reached with very few hops, reflecting the global scale of tier-1 ISP networks. This study of inter-domain network topologies has shaped various approaches to network simulation and protocol design [78, 93].

Lixin Gao [49] studied Route Views paths to determine the relationship between pairs of ASes. The relationship between ASes, whether one is a customer of another or two ASes are comparably-sized peers, can be inferred by applying two common, high-level routing policies that ASes use to choose AS paths: prefer-customer and no-valley. The *prefer customer* routing policy represents each ISP's preference for the routes made available by their customers over routes made available by peers over routes made available by providers. The *no-valley* ("valley-free") routing policy represents that customers and peers do not provide *transit*: connectivity to the rest of the Internet. In BGP, customers do not advertise routes to the rest of the Internet to their providers and providers do not accept such routes from customers. The exchange of traffic between comparably-sized peer ASes is usually limited to reaching customers of the other ISP. The no-valley policy means that AS paths consist of customer to provider edges, followed by at most one peer to peer edge, followed by provider to customer edges. Conceptually, traffic flows "uphill" from customers to their providers to some "peak," then

[2] These ISPs need not all be present in Oregon. Since the work in this dissertation was completed, the Route Views server in Oregon has grown to connect to 60 ISPs and the project overall connects to 88.

flows back “downhill” from providers to customers until it reaches a destination, with no “valleys” in between. With sufficiently many observed paths, the edges between ASes in the inter-domain topology can be classified as customer-to-provider (uphill), provider-to-customer (downhill), or peer-to-peer (neither).

3.2.3 Limitations

Route Views provides a partial view of inter-domain connectivity because every BGP router in the Internet may have a different view of connectivity. Further, the design of BGP allows each router to choose different forwarding paths, even when given the same input and managed by the same ISP [127].

Mao *et al.* [75] recently uncovered a further limitation: the AS path listed in a BGP table may not match the forwarding path used. This inconsistency occurs even when perfect BGP information is available. Inferences based on the apparent inter-domain topology and routing policies may be questionable if they are not matched by physical connectivity.

Global routing information can provide a list of destination prefixes that are likely to be routed differently, but not all addresses within globally-routed prefixes will be routed identically. Because each IP address prefix that is advertised globally occupies routing table space on every BGP-speaking router in the Internet, operators of those routers apply social pressure to limit how many prefixes are advertised. ISPs are encouraged to aggregate (merge) adjacent prefixes that appear to the rest of the network to be routed identically [12, 48]. This aggregation is most common for prefixes originated by an ISP because aggregating the prefixes originated by another ISP creates a less-specific route that is less likely to be selected by other ISPs and perhaps less profitable.

3.3 DNS Naming Information

Naming is part of the debugging interface of the ISP networks. ISPs can, but do not always, associate human-readable names with router addresses to allow users and other ISPs to better understand what geographic paths their packets traverse. This information can be extracted to add structure to measured network maps.

3.3.1 How DNS Names Match IP Addresses

The Domain Name System (DNS) is a distributed database that matches IP addresses with hierarchically constructed names. This binding can be queried in both directions: finding the IP address for a name (forward) and finding the name for an IP address (reverse lookup). The description that follows describes the hierarchical design and reverse lookup at a high level. For a more precise discussion, see Stevens [118].

The DNS exploits hierarchy, and, like the Internet, features distributed administration. To find the IP address for a name, for example, `fretless.cs.washington.edu`, a *resolver* first queries a server for the top-level domain, `.edu`. This server knows about every immediate sub-domain of `.edu`, so can respond with contact information for the server responsible for `.washington.edu`. This server will respond with contact information for the server responsible for `.cs.washington.edu`, which can answer the query for the IP address corresponding to `fretless`. This hierarchical design allows hosts to be added and removed without updating a centralized database.

A *reverse lookup*, finding the name for an IP address, uses a similar procedure. The resolver reverses the IP address and appends a special domain, `.in-addr.arpa`, to form the query. For example, to reverse lookup the IP address 1.2.3.4, the resolver will ask for information about `4.3.2.1.in-addr.arpa`. Hierarchy is still possible: recall the BGP table in Figure 2.4 in which 1.2.3.0/24 was an advertised prefix; the origin AS for that prefix runs a DNS server responsible for the `.3.2.1.in-addr.arpa` sub-domain. Usually, this means that routers owned by an ISP have clear names.

3.3.2 Prior Work Interpreting DNS Names

Padmanabhan and Subramanian investigate the use of DNS names to discover the locations of Internet hosts and the geography of Internet paths [91, 122]. They design and evaluate a tool, *GeoTrack*, which uses a list of location-identifying fragments to assign a location to a router name. Their research extends this router-location approach to infer the location of different end-hosts in the network with mixed results compared to alternate, latency-based approaches [91]. In follow-on work, they applied the name-based technique to understand how geographically circuitous Internet paths were, providing initial insight into the prevalence of peering policies such as late- and early-exit without measuring a complete topology [122]. Most significantly for building structured, router-level ISP topologies, this work opened up the possibility of more completely decoding DNS names to understand even more of the information made available by ISPs.

3.3.3 Limitations

Because the IP address prefix used on a peering link belongs to one ISP or the other, DNS is not always a reliable indicator of the owner of a router. The administrator of the `.in-addr.arpa` domain associated with the peering link prefix has to decide what name to assign to the "other" end of the link that belongs to the other ISP. One approach is the practice of Level3, a tier-1 ISP in the United States: it assigns the name `unknown.level3.net` to every address it owns that is not part of the Level3 network. The `.level3.net` suffix would suggest that the router is owned by Level3, even when it is not. When I use names to find ISP boundaries, ISPs using such names are treated differently; for example, the "unknown" names are discarded.

More subtly, ISP may assign an approximately-correct name to the other end of the peering link. My favorite such name is `att-gw.sea.sprint.net`. AT&T commonly names the end of the peering link owned by the other ISP in this way. The `gw` stands for gateway and the `sea` for Seattle. This naming practice is useful for observing that a path has left AT&T, but is not necessarily useful for discovering which ISP it has entered because the name is not quite right. Sprint uses the name suffix `sprintlink.net` for its routers, and has an entirely different naming convention—one which I use as an example in Section 4.2.

DNS names are not universally used. They may be given only to some router interface addresses within an ISP, and are missing from some ISPs altogether, particularly ISPs in Korea and China. It is an open question how many ISPs can (and cannot) be understood through the DNS information they provide.

Finally, the use of DNS names to understand the structure of the topology and the owners of different routers has not yet been well-validated. Using DNS names requires relying on an interpretation of optional information published by ISPs. ISPs are not penalized for out-of-date or incorrect router interface DNS names, unlike both BGP information, which must be correct enough for correct forwarding, and traceroute information, which is safer to block than to corrupt. Other information, including performance information measured by traceroute, can be used to detect some incorrect DNS names.

3.3.4 Summary

Prior work has used what information is available through traceroute, Route Views, and DNS names to understand aspects of the structure of the Internet. I will show how these data sources can be combined to measure structured, router-level ISP topologies and infer routing policies. For example, information from Route Views will help to guide the focus on an ISP at a time. Information from DNS can be corrected by the latency measurements from traceroute to find errors in the measured topology. Finally, information from DNS names complements origin AS information from routing tables to determine which ISP owns each router.

Chapter 4

ISP TOPOLOGY MAPPING

To measure structured, router-level ISP topologies efficiently and accurately, I developed several techniques that can be used by anyone on the Internet. Together, the implementations of these techniques form the *Rocketfuel* ISP topology mapping engine.

The accuracy of these techniques has two components: completeness and correctness. *Completeness* is the fraction of links and routers found. Using many vantage points, hundreds of public traceroute servers, is intended to help completeness by exposing many paths through the network. *Correctness* is whether the information that was recovered is true: a combination of correct alias resolution, avoidance of false links, and correct assignment of role and location to routers. Extra traceroute measurements can help completeness, because they may discover a new links or routers, but they can compromise correctness because routers and links may move in the extra time needed to collect the measurements.

A unifying principle underlying these techniques is to focus on the measurements that are most likely to contribute new information. The number of measurements required to discover every link is much smaller than the product of vantage points and destinations. Similarly, the number of likely IP address aliases is much smaller than all-pairs because IP addresses in different locations are not aliases; the location of an IP address can be estimated before alias resolution. Recognizing redundant measurements and exploiting parallelism can reduce the time to complete a map. This speed in network mapping yields improved correctness because the network is not allowed time to change. The balance between choosing only the measurements most likely to contribute information and choosing enough measurements to collect a complete map is a tradeoff in which I choose to collect as many measurements as are practical.

This chapter is organized around three main problems. The first problem is how to map ISP networks using traceroute efficiently: selecting traceroutes that have the potential to add new information. The second is how to resolve IP address aliases to form a router-level map. The third problem is how to recover the geographic structure of the topology by interpreting hints in the DNS names of router interfaces. This chapter provides the design and implementation of solutions to these problems, deferring an analysis of their effectiveness to Chapter 5.

4.1 Choosing Traceroutes for Efficient ISP Topology Mapping

The approaches to Internet mapping described in Section 3.1.2 use a dedicated platform of tens of vantage points and run traceroute to as many destinations as they can. The dedicated platform has an advantage: it can run many traceroutes in a short period of time, but also a significant disadvantage: the number of vantage points is limited. Although these projects have measured to each of the 120,000 prefixes that are globally-visible unique destinations in BGP, and can traceroute to more than a million destinations, the resulting maps may not be particularly accurate because of the inherent limitations of traceroute from few vantage points.

I choose a different approach: I focus on one ISP at a time and use hundreds of public traceroute servers as vantage points. An ISP is the unit of the network that is designed by a single organization; it runs a single routing protocol and has a single routing policy when interacting with neighbors. As such, detailed information about an individual ISP can be quite useful in research. Focusing on a relatively small component of the network makes it possible to take only those traceroutes likely to traverse new paths through that network. These traceroutes are fewer than the all-to-all approaches used previously, and, because they are fewer, this approach enables the use of public traceroute servers as vantage points. Public traceroute servers have two salient features: they are plentiful and they support only limited traceroutes. It would take 125 days to use a public traceroute server to trace to the 120,000 globally visible prefixes in BGP, when using a rate limit given to me by the administrator of one traceroute server.[1]

My insight is that routing information and prior measurements can help select the measurements that are likely to provide new information. One technique I devise, *directed probing*, uses BGP routing information to choose only the relatively few traceroutes that are likely to transit the ISP being mapped so that effort is not wasted on other parts of the Internet. A second set of techniques, *path reductions*, suppress traceroutes that are likely to yield paths through the ISP network that have been already been traversed. Directed probing and path reductions reduce the number of traces required to map an ISP by three orders of magnitude compared to a brute-force, all-to-all approach, without significantly sacrificing the accuracy of the result.

4.1.1 Directed Probing of Specific ISP Networks

Directed probing selects traceroutes that are likely to transit a selected ISP network. Because the Internet comprises so many ISPs, most of the possible traceroutes are unlikely to traverse any particular ISP and should be skipped. BGP routing information provides a means to predict, before traceroutes are taken, whether a traceroute is likely to traverse any chosen ISP.

Routers running BGP maintain a routing table that maps destination IP prefixes to the path of Autonomous Systems (ASes; each AS is roughly equivalent to an ISP) traversed to reach that destination. With this information, each router could predict, for the packets it forwards, whether that packet will traverse a particular ISP. Unfortunately, these BGP routing tables are rarely available on end-hosts, and are generally not available on the measurement platform of public traceroute servers.

I use the Route Views BGP archive described in Section 3.2 to provide BGP routing information. Recall from Section 3.2.3 that the Route Views archive provides only partial views of inter-domain routing, and the routers participating in Route Views are distinct from the traceroute servers I use. As a result, directed probing may choose unnecessary measurements or miss potentially useful ones. However, I will show that the maps I measure are sufficiently accurate.

I now show how to use partial BGP views to identify three classes of traceroutes that are likely to transit a chosen ISP network. In this example, I use the sample BGP table snippet in Figure 2.4 on page 12 to map AS number 7 by selecting three classes of traceroutes.

Class 1: Traceroutes to dependent prefixes. I call prefixes originated by the ISP or one of its singly-homed customers *dependent prefixes*. All traceroutes to dependent prefixes from any vantage point transit the ISP because there are no other paths. Dependent prefixes can be readily identified from the BGP table: all AS-paths for the prefix contain the number of the AS being mapped. In Figure 2.4, `4.5.0.0/16` is a dependent prefix of AS 7. Restated, `4.5.0.0/16` is downstream of AS 7 in every path.

Two types of traceroute may violate this assumption. First, traceroutes from a vantage point within the dependent prefix may not need to transit the ISP being mapped. Second, if the prefix is not truly dependent—it only appears to be from the limited BGP views available—traceroutes may measure a path that does not traverse the ISP.

Class 2: Traceroutes from insiders. I call a traceroute server located in a dependent prefix an insider. Traceroutes from insiders to any of the globally visible prefixes in BGP should transit the ISP. Again, if the prefix is

[1] The rate given by this administrator was one traceroute per 1.5 minutes. I considered this to be too aggressive for other servers that were less well-provisioned and used a rate of one traceroute per 5 minutes during mapping. This would take at least 417 days to complete a map. I used the limit provided by the administrator above to avoid making the problem arbitrarily more difficult than necessary: 125 days is plenty long for the network to change.

misclassified as dependent, or the source and destination are both within the same customer network, the trace may not traverse the ISP.

Class 3: Up/down traceroutes. Other traceroutes that are likely to transit the ISP based on some AS-path but are not matched by either rule above are called *up/down traces*. "Up/down" represents that a traceroute server is in an "upstream" AS and the destination in a "downstream" AS that can be identified by BGP paths. In Figure 2.4, a traceroute from a server in AS 11 to `1.2.3.0/24` is an up/down trace when mapping AS 7. Some insider and dependent prefix traces will also be matched by the up/down rule; up/down traces are those that match this rule and no other. This "other" class of traceroutes may mispredict because of incomplete BGP information: different routers within the same AS may make different decisions [127].

My approach is to skip traceroutes that match none of these three criteria. Incomplete information in BGP tables, dynamic routing changes, and multiple possible paths lead to two kinds of errors, which I quantify in Section 5.2.1. First, executed traceroutes that do not traverse the ISP (false positives) sacrifice speed, but not completeness: the traceroute simply should not have been taken. Second, traceroutes that would have transited the ISP network, but were skipped because limited BGP data did not include the true path (false negatives), may represent a loss in completeness. False negatives may not always compromise completeness because traceroutes that were not taken may traverse the same links seen by traceroutes that were. False negatives could be eliminated by exhaustive measurement, but that approach is impractical.

4.1.2 Path Reductions Discard Redundant Measurements

Many traceroutes chosen by directed probing will take overlapping paths inside the ISP network. Path reductions identify these traceroutes so that they can be skipped.

Two traceroutes that enter (ingress) and exit (egress) the network at the same points are likely to take the same path through the ISP, and if so, this path needs to be measured only once. Path selection is consistent because forwarding decisions at each router are usually based only on the destination address. Each traceroute provides information to help predict where a future trace will enter and leave the ISP network. I list three techniques based on common properties of IP routing to predict which paths are likely to share ingress and egress points.

4.1.2.1 Ingress reduction

Traceroutes from a vantage point may enter the ISP at a deterministic *ingress*. When many vantage points share an ingress (all deterministically use the same ingress), this presents an opportunity for avoiding unnecessary measurement. Figure 4.1a illustrates the sharing of an ingress. A traceroute from T2 would discover the same information as a traceroute from T1 to the same destination; one of these is redundant and can be skipped.

Each traceroute provides an observed ingress point used by the vantage point to guide the prediction of sharing. I elect the ingress point seen in the majority of the traceroutes from a vantage point to be its predicted ingress. Only when most of the traceroutes from a vantage point use the same ingress do I apply ingress reduction. The vantage points that appear to share an ingress are equivalent: traces from any of these are likely to discover the same information.

Traceroutes that do not enter the ISP at the expected ingress sacrifice either efficiency or completeness. Assume without any loss of generality that T1's trace to a destination is taken first. Because traces are taken in sequence, there are two cases when an ingress prediction is incorrect: T1's ingress is predicted incorrectly (and T2's may or may not be), or T1's ingress is predicted correctly but T2's is not. I can detect when T1's traceroute does not use the predicted ingress, and try T2 to preserve completeness at the cost of efficiency. If T1's ingress is predicted correctly, I will not try T2; the effect is that some completeness may be sacrificed if the prediction is incorrect.

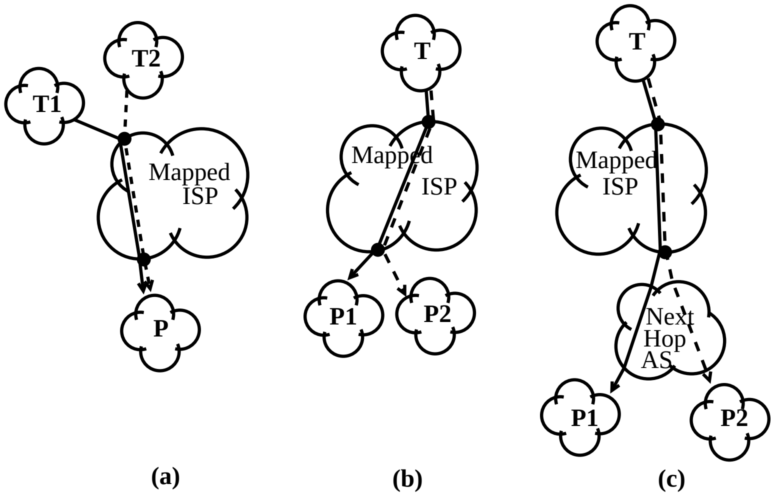


Figure 4.1: Path reductions find traceroutes likely to traverse the same path through an ISP. (a) *Ingress Reduction*: Only one traceroute needs to be taken per destination when two vantage points (T's) share an ingress. (b) *Egress Reduction*: Only one traceroute needs to be taken when two dependent prefixes (P's) share an egress router. (c) *Next-hop AS Reduction*: Only one traceroute needs to be taken if two prefixes are reached through the same next-hop AS.

4.1.2.2 Egress reduction

Conversely, dependent prefixes owned by the ISP and that share an egress router are equivalent. Figure 4.1b illustrates the sharing of an egress router. I find the egress router for each dependent prefix by running traceroute; the last router traversed within the ISP is the egress for that dependent prefix.

The egress router that traceroute can discover from a single vantage point may not be the only egress router used to reach dependent prefixes that are multi-homed within an ISP and do not have an AS. I expect that such dependent prefixes are rare because geographically large enterprise networks are unlikely to rely on only one ISP for connectivity. That is, large networks that multi-home within a single ISP are likely to also connect to other ISPs and are not dependent prefixes. Networks that connect only to one ISP may still have an AS for load-sharing and reliability [119]; such prefixes are treated using the next-hop AS reduction rule below. Nevertheless, a false egress router binding does not necessarily sacrifice completeness because other dependent prefixes may connect to the same egress routers.

4.1.2.3 Next-hop AS reduction

When reaching prefixes outside the ISP, the path usually depends only on the next-hop AS, and not on the specific destination prefix. Figure 4.1c illustrates prefixes reached through the same next-hop AS. While egress reduction finds prefixes that share an egress router, next-hop AS reduction finds prefixes that share a next-hop AS.

The difference between next-hop AS and egress reductions is subtle because both predict where a trace will leave the ISP. The difference is that the exit point for a dependent prefix is an egress router, while the exit point for a prefix in another AS is identified by the next hop AS. As a result, egress reduction applies to traces to dependent prefixes owned by the ISP, while next-hop AS reduction applies to insider traces, up/down traces, and traces to dependent prefixes in customers having an AS.

Next-hop AS reduction does not assign an egress router to each destination prefix. Each vantage point may observe a different early-exit point when reaching the next-hop AS. That is, when reaching a next-hop AS, multi-homing is common, while when reaching a prefix originated by the ISP, multi-homing is not. Next-hop AS reduction, as a result, groups destination prefixes by the next-hop AS, while egress reduction groups destination prefixes by egress router.

These three path reductions predict where traceroutes will enter and leave the ISP. When a path between these ingress and egress points has already been discovered, the traceroute is skipped. If the prediction is observed to be false (an unexpected ingress or egress was observed in a traceroute), the traceroute is repeated using other servers until each ingress to egress path has been measured.

4.1.3 Implementation: Rocketfuel

Figure 4.2 shows the data flow of the Rocketfuel ISP mapping engine. The Rocketfuel implementation uses an SQL database to store intermediate results in a blackboard architecture [87, 88]: the database provides a means for independent processes to communicate. The database supports SQL queries to answer simple questions and helps to integrate new analysis modules.

I now describe the modules in Figure 4.2, in topological order, but defer alias resolution and location inference to the following sections.

The directed probing module takes as input the list of vantage points and BGP tables from Route Views [80] to find the dependent prefix, insider, and up/down traces. It creates first a list of jobs to execute: these jobs are represented by tuples that include the vantage point, the destination prefix, and some information about why that traceroute was selected. It also creates a list of dependent prefixes that the egress discovery module will use.

Egress discovery is the process of finding the egress routers for dependent prefixes—information used for egress reduction. Because dependent prefixes may have been aggregated by the ISP, to find the egress routers, I traceroute to a randomly chosen address in each /24 in each dependent prefix. The egress router is the last address in the ISP seen by traceroute. (Technically, this “egress router” is just an egress router address, not yet a router after alias resolution. Using only one source makes finding aliases less likely.) This process may discover several egress routers for dependent prefixes larger than a /24. Running traceroute to /24s does not discover *all* ISP egress routers: using /30s instead discovers 8% more egresses, depending on the ISP, but requires 64 times as many traceroutes. (I present more detail in Section 5.2.2.2.)

The path reductions module takes the list of jobs generated by directed probing, replaces dependent prefixes with the addresses of egress routers, and removes jobs likely to be redundant with traceroutes already collected. The path reductions module re-computes the jobs list as completed traceroutes show which ingresses are used by each vantage point. The list of jobs after path reduction includes traces directly to egress routers; after the traceroute leaves the ISP, no information is provided, so there is little reason to run traceroute beyond the egress router. The path reductions module also tracks which jobs have been completed; jobs are removed if they would be redundant with those completed. A collected trace in which the ingress point prediction was incorrect will not cause the job to be marked completed; the path reductions module will not remove jobs redundant with the trace attempted, only jobs redundant with traces collected.

The traceroute execution module handles some of the complexities of using public traceroute servers: load-limiting and load-balancing. Load is distributed across destinations by deterministically shuffling the job list, implemented by sorting by the MD5 hash [109] of each job. A five minute pause between accesses to the same traceroute server avoids overloading it: although one friendly traceroute server administrator suggested 1.5 minutes was sufficient, I wanted the extra safety margin in case other administrators were less tolerant or had servers with less capacity. Traceroutes to the same destination prefix are not executed simultaneously to avoid hot-spots.

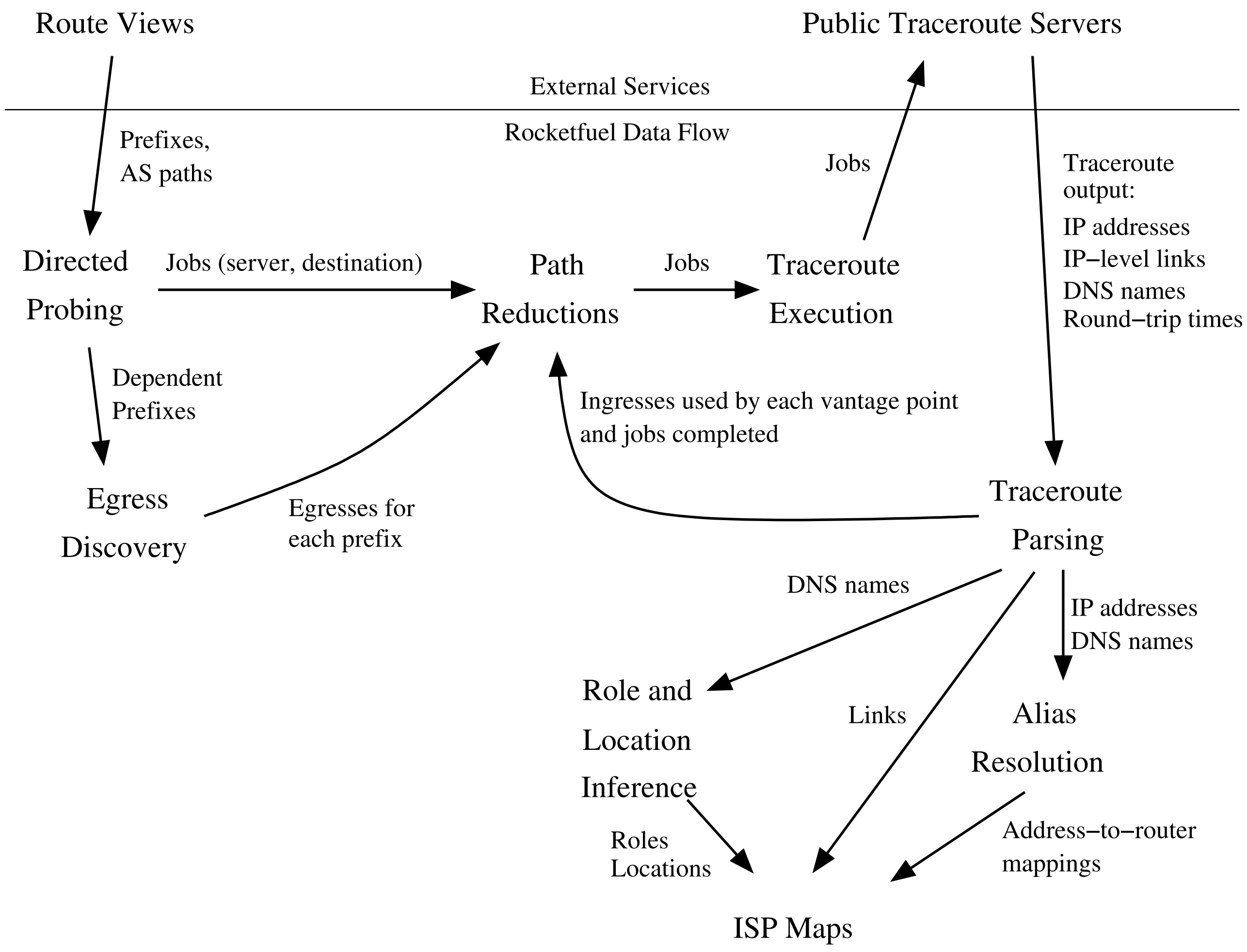


Figure 4.2: The flow of data through the Rocketfuel topology mapping engine. A database (not shown) provides temporary storage of intermediate results. Route Views and public traceroute servers exist outside the Rocketfuel system; they may be replaced by other sources of BGP and traceroute measurements. The data sent along each edge annotates the edge. Data before traceroute execution consists primarily of jobs to be run; after traceroute execution, data consists primarily of topology data.

The traceroute parser extracts topology information from various formats of traceroute output, removing presentation mark-up like headers, tables, and graphics. The topology information that results includes IP addresses, IP-address to IP-address links, and the DNS names associated with the IP addresses. The traceroute parser also determines whether the observed trace traversed the expected path and marks the job completed if so.

In summary, the Rocketfuel ISP mapping engine is a modular system based on a blackboard architecture. Each analysis or measurement module contributes intermediate results that determine which future measurements will be collected.

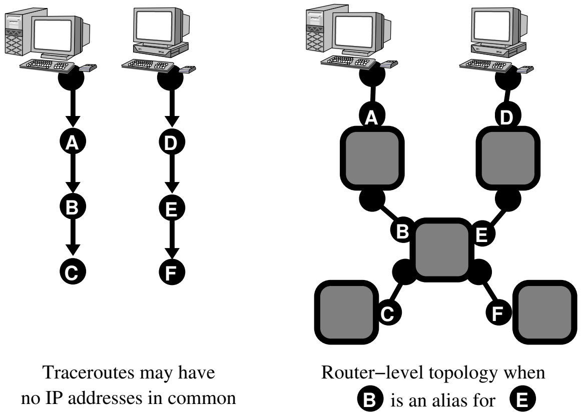

A
B
C
D
E
F
A
D
B
E
C
F
Traceroutes may have no IP addresses in common
Router−level topology when B is an alias for E

Figure 4.3: The problem of alias resolution: recognizing interfaces that belong to the same router. Boxes represent routers and circles represent interfaces. Traceroute lists input interface addresses from paths (left). Alias resolution clusters interfaces into routers to reveal the true topology. Interfaces B and E are aliases (right).

4.2 Resolving IP Aliases to Routers

Each traceroute provides a list of IP addresses that represent router interfaces. For a correct router-level map, the IP addresses that belong to the same router, called *aliases*, must be resolved. Refer to Figure 3.1 on page 16 for sample traceroute output and to Figure 3.3 for an example topology that is ambiguous without alias resolution. Prior techniques for alias resolution did not resolve obvious aliases, including some within my university network. In response, I developed a new, pair-wise test for aliases that uses router identification hints such as the IP identifier, rate-limiting, and TTL values.

Alias resolution is interesting because the problem is both caused and solved by somewhat esoteric design choices in how error messages are generated by routers. Traceroute lists the source addresses of the “Time exceeded” ICMP error messages; these addresses represent the link interfaces on the routers that received traceroute probe packets [118, Section 8.3]. Figure 4.3 illustrates the problem. If the different addresses that represent the same router cannot be resolved, then the result is a different topology with more routers and links.

4.2.1 Alias Resolution Techniques

The standard technique for alias resolution was introduced by Pansiot and Grad [92] and refined by Govindan and Tangmunarunkit as part of the Mercator project [55]. It detects aliases by sending traceroute-like probes (to a high-numbered UDP port but with a TTL of 255) directly to every discovered IP address. Responses from aliases will have the same source address, but only if the router is configured to send the “UDP port unreachable” response with the address of the outgoing interface as the source address as required by the standard [9]. This technique is efficient in that it requires only one message to each IP address, but it misses many aliases. This may be because some routers are not configured to respond with the outgoing interface address as the source address.

My approach to alias resolution combines the source-address-based alias resolution approach with new pairwise tests. These techniques try to collect evidence that the IP addresses belong to the same router by looking for features that are centrally applied.

I look primarily for nearby IP identifier values. The IP identifier is a 16-bit field in the IP header that helps uniquely identify a packet for reassembly after fragmentation [100]. As such, it is commonly implemented

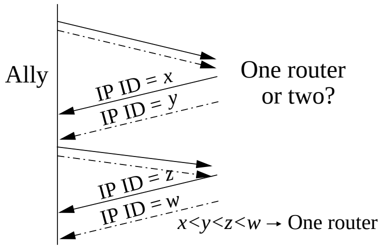


Figure 4.4: Alias resolution using IP identifiers. A solid arrow represents messages to and from one IP address, a dotted arrow messages to the other.

using a counter that is incremented after generating a packet.[2] This implies that packets sent consecutively from the same router will have consecutive IP identifiers, a feature that can be exploited to test for aliases. Routers do not appear to generate packets at a rate approaching 2^{16} packets per second; if they did, it would be difficult to determine whether two addresses respond with the same counter. Different routers do, however, generate packets at different rates, perhaps depending on how much control traffic they must generate to participate in a network.

Figure 4.4 illustrates the procedure for resolving aliases by IP identifier. My tool for alias resolution, Ally, sends a probe packet similar to Mercator's to each of the two potential aliases. The port unreachable responses include the IP identifiers x and y. Ally then sends a third and fourth packet to the potential aliases to collect identifiers z and w. If $x < y < z < w$, and $w - x$ is small, the addresses are likely aliases. In practice, some tolerance is allowed for reordering in the network. As an optimization, if $|x - y| > 200$, the aliases are disqualified and the third and fourth packets are not sent. In-order IP identifiers are evidence of a single counter: the addresses are likely aliases. As I show in Section 5.3.1, this test is reasonably accurate.

Different routers may, with small probability, have nearby identifiers. To remove the resulting false positives, I repeat the IP identifier test after some time to verify the alias.

The network maps presented in the rest of this dissertation were generated using a three-packet technique, without the w packet. The fourth packet reduces the false positive rate of the test. I evaluate the four-packet version in Sections 5.3 and 5.4. Because different routers change their IP identifiers at different rates, the four-packet test establishes that the two addresses have counters with similar value and rate of change, while the earlier three-packet test only demonstrated that the counters had similar value.

Some routers are configured to limit the rate at which they generate the port unreachable messages Ally solicits [32]. If only the first probe packet solicits a response, the probe destinations are reordered and two probes are sent again after five seconds. If again only the first probe packet solicits a response, this time to the packet for the other address, the rate-limiting test detects a match. This test is not effective alone because it relies on a specific pattern of loss that can occur by chance (for example, due to congestion).[3] When two addresses appear to be rate-limited aliases, I also require the identifiers to differ by less than 1,000 for a match.

[2] The common implementation of the IP identifier with a counter is considered well-known but not specified [14, 15, 20]. I have not observed routers that use random identifiers or implement the counter in least-significant-byte order, though some do not set the IP ID at all; such routers are not amenable to this technique.

[3] Rate-limiting routers usually replied with the same source address; although peculiar, ICMP rate limiting did not appear to be an accurate test on its own.

4.2.2 Efficiency in Alias Resolution

The process of alias resolution is complete when all likely pairs of IP addresses are resolved as aliases, not aliases, or unresponsive. Pairwise testing of all addresses discovered by traceroute to see if they are aliases is inefficient when there are many addresses. My approach is to test, out of all pairs of addresses, only those likely to be aliases. This approach may sacrifice completeness to gain efficiency (and thus correctness by denying the network time to change). To reduce the search space from all pairs of IP addresses to a more manageable list of likely candidates, I apply a heuristic that uses the return TTL in responses.

Router IP addresses with replies that have nearby return TTLs may also be aliases. The *return TTL* is the TTL remaining in the outer header of the port-unreachable response: a function of the initial TTL at the router and the length of the path back to the host. The *outgoing TTL* is the TTL remaining in the header of the original probe message when it reached the router, returned as part of the header embedded in the port-unreachable response. (The header of the original message is included in each error message so that the machine that receives the error message can determine which process should receive it.) The outgoing TTL depends on the initial TTL of the host and the length of the path to the router IP address. The return TTL is more correlated with aliasing: in initial tests from a single vantage point, of the 16,000 aliases I found, 94% matched the return TTL, while only 80% matched the outgoing TTL.[4] To exploit this, I group addresses by the TTL of their responses and test pairs with nearby return TTLs, starting with those having equal TTL, then those within 1, and so on, until no new aliases are found. Because network routing may change while TTLs are being discovered, some tolerance is required in testing pairs with slightly different TTL values.

4.3 Decoding DNS Names to Find Structure

In this section, I describe how to use DNS names to solve two problems in providing structure in network maps. First, DNS names can help find the boundaries between ISPs. Second, many DNS names expose the geographic location and role of each router. I now describe each problem in more detail.

4.3.1 Drawing a Boundary Around Each ISP

To identify the routers that belong to the ISP, I use DNS information when it is available. Router DNS names indicate ownership more accurately than the IP address space advertised by the AS in three settings.

Customer routers may be numbered from the ISP's address space. DNS names help to locate the edge of an ISP network because the names of customer routers are typically given by the customer. For example, a path may include several routers from `att.net` followed by some in `example.com` without leaving AT&T's address space. Some ISPs use a special naming convention for neighboring domain routers to denote the network edge. For example, Sprint names customer routers `sl-neighborname.sprintlink.net`, which is different from Sprint's internal router naming convention.

Conversely, a router interface may have an IP address owned by the ISP to which it connects or by a third ISP. IP addresses belong to a subnet that represents a network link, as described in Section 2.1.3. Both ends of a private peering link between ISPs are numbered from the same prefix. Only one ISP owns the address space used for a peering link; the prefix is too small to be worth advertising on its own. Peering links at exchange points may have subnets owned by a third party; the result is the same: individual IP addresses are an unreliable indicator of the ownership of a router. (Alias resolution provides another solution, but only if enough interfaces within the ISP are found that the correct AS for a router can be "elected.")

[4] Fewer match the outgoing TTL for reasons that are subtle and unimportant for this discussion. Because routers make forwarding decisions based on destination addresses and store prefixes instead of individual addresses, an outgoing probe may go first to the neighbor of the destination router—the neighbor having an address on the same prefix as the destination address. This different path means the TTL may be decremented more.

Finally, DNS names help prune out cable modems, DSL, and dialup modem pools belonging to the same organization as the ISP, and hence numbered from the same IP address space. Frequently, these parts of an ISP network are administered by a different organization and act as customers. I resort to the IP address space criterion for routers with no DNS names (I observed very few of these), with the constraint that all routers belonging to the ISP must be contiguous in the traceroute output.

4.3.2 Exposing Geography

The most important information for understanding the structure of router-level ISP maps is the location of each router. This information is often embedded in DNS names. Although few DNS names identify a precise building, many identify the metropolitan area where a router is located. A DNS name may also indicate the role a router fills, such as whether it is a backbone router or an access router. The ISPs I studied have a naming convention for their routers that provides this information in a consistent format. For example, `sl-bb11-nyc-3-0.sprintlink.net` is a Sprint backbone (bb) router in New York City (nyc), and `p4-0-0-0.r01.miamfl01.us.bb.verio.net` is a Verio backbone (bb) router in Miami, Florida (miamfl). I discover the naming convention of the ISP by browsing through the list of router names I gather. For some ISPs, I started with city codes from the GeoTrack database [91].

Relying on DNS names to provide geography has limitations. Some routers have no DNS names or their names lack location information. I infer the location of such routers from that of its neighbors, when unambiguous. A router with unknown location may be connected to routers in different locations; this prohibits inferring a location for that router so the assignment of geography may be incomplete. Some router interfaces have misconfigured or obsolete DNS names: while investigating the accuracy of alias resolution, I observed a few address pairs where the addresses had names that suggested they were in different POPs, yet the alias resolution tests consistently claimed that the addresses were indeed aliases. Limited sanity-checking can ensure that only physically possible assignments are made, given the observed latency of communicating with an address. I apply this sanity-checking and evaluate the completeness of inferred geographic locations in Section 5.5.

4.3.3 Implementation: The undns Library

The `undns` library is my engine for extracting the information embedded in router names. Each ISP has a *naming convention* represented by regular expressions. Some naming conventions are more easily expressed using several regular expressions. When a regular expressions matches, the substrings (parenthesized expressions) are extracted and looked up in a table that matches city abbreviations to city names and router type tags to the canonical router types (backbone, gateway, customer). Substrings can be extracted without this table, supporting an alias resolution technique in which only the identifier in the name matters.

The use of regular expressions allows two optimizations that increase the throughput of decoding. First, expressions are pre-compiled at load time which speeds the matching process. Second, general regular expressions guard more specific regular expressions, ensuring, for example, that a router name includes `.att.net` before comparing it to each of the AT&T naming convention expressions.

Figure 4.5 shows a sample naming convention from `undns`. The Autonomous System Number (ASN) for BellSouth, 6389, appears first. Names that match this convention are considered to be part of that AS. Only names that end in `.bellsouth.net` are processed further. BellSouth uses three-letter airport codes to represent locations, these codes are placed in the second (`loc=2`) parenthesized subexpression, `([a-z]{3})`. Because the airport tag is a common convention, a dictionary of airport tags is kept in a separate file and included. This simple rule does not include annotations for discovering aliases and router types, but they are expressed in a similar style.

Using an ISP-specific rule set improves the accuracy of the result. For example, “Vienna” in a router name may represent Austria or Virginia, and it is reasonably easy to tell which, given the rest of the topology and the

```
6389 \.bellsouth\.net\$ {
  (ixc|axr)[0-9]{2}([a-z]{3})[0-9-]*\.bellsouth\.net$ loc=2 {
    #include "keys/three-letter-airport"
    mem "Memphis, TN"
    bhm "Birmingham, AL"
  };
}
```

Figure 4.5: A sample naming convention rule from undns.

area served by the ISP. Similar ambiguity applies to Rochester (New York or Minnesota), London (England or Canada), and several other city names.

I took great care to make the location mapping complete and correct. While processing traces, my analysis flags names that match the router name pattern of ISP backbone routers but have undefined locations. I manually populate the tables of location codes based on this output until complete. Not all routers have location codes (Sprint customer routers, for example); I infer the location of such routers from that of their neighbors. A problem with trusting DNS names for router location is that routers may be improperly assigned to locations. The DNS name may be incorrect or stale, or the location inferred may be incorrect. For example, one IP address had the location tag “ALT,” which is the three-letter airport abbreviation for Alenquer, Brazil. This router was in Atlanta, and the name was simply “ATL” mis-typed. I discover bad location matches by flagging physically impossible link latencies, limited by speed of light in fiber (in this example, Georgia to Brazil in 1 ms). Later analyses ignore the trace until the router location has been corrected.

4.4 Limitations

The techniques presented in this chapter worked well for the ISPs I studied, but the primitives they rely upon are not universally available and may even be disabled in the future. For each of the following information sources, I describe why I believe they will not be disabled by ISPs and, for some, how outsiders might collect the same information if the sources I used were disabled.

Traceroute through the ISP can be disabled. One mechanism would be not to decrement the TTL as a packet traverses the network, which is safe when, for example, MPLS creates loop-free label-switched paths. Routers can also be configured not to send time-exceeded error messages back to the source—such routers are “anonymous.” Recent research has investigated how to form a topology when anonymous routers are present [134], but techniques rely on having relatively few anonymous routers, not an entirely anonymous ISP.

Traceroute to ISP network prefixes could be disabled. Some ISPs filter incoming traffic destined for their internal network addresses. Filtering helps avoid the possibility that others might exploit vulnerabilities in router software. Although Rocketfuel uses traceroutes *through* the network, and traceroutes to egress routers could be replaced with traceroutes beyond the egress routers, blocking traffic to router addresses would block alias resolution probes. DNS names are one way to provide alias resolution if probes fail. I investigated an alternate approach based on the IP-level connectivity graph, with limited success [114].

DNS names could be made opaque or removed. The value of DNS names makes this unlikely—ISPs can use DNS names to help locate problems and their geographic tags show off the size of the network. Few ISPs in China and Korea, however, use DNS names. It is unclear whether these ISPs decided to hide the network topology or simply did not see the value of router DNS names. Padmanabhan and Subramanian [91] provide a means of estimating geographic location using latency performance measurements instead of DNS names. This approach requires a geographically diverse platform and may not be able to to determine the precise location of routers that are slow to respond to probes.

Route Views could cease to be funded or be abandoned by ISPs. It is not possible for an individual ISP to "hide" from Route Views, but the data source could still become unavailable. It has, however, proven quite useful to researchers and operators, so its disappearance is unlikely. An alternative to Route Views is the direct BGP feed to dedicated measurement platforms. Any network that hosts a PlanetLab machine may be able to collect the local view of BGP information. Without any BGP information at all, many of the techniques described in this dissertation would be ineffective.

If all of the information sources I used were disabled—in particular both DNS and the ability to send packets directly to network addresses—it would be particularly difficult to determine the locations and roles of individual routers. That is, an ISP could conceal its topology if it wanted to. That they do not hide their topologies suggests that it is not worth their effort to do so.

4.5 Summary

The methods described in this chapter were designed to map the topologies of selected ISPs. My intent was to recover the structures that are designed, and this focus on an ISP at a time meant that I was able to collect accurate and useful maps efficiently. The philosophy underlying these techniques is to focus on the measurements most likely to contribute new information.

Chapter 5

ISP TOPOLOGY MAPPING EVALUATION

This chapter evaluates the Rocketfuel techniques. I evaluate each of the techniques from Chapter 4, starting with the efficiency with which the traceroutes were collected, the accuracy of individual alias resolution methods, the efficiency with which alias resolution can be run, and the plausibility of geographic locations inferred using DNS names. I describe the results of comparisons that give confidence in the overall accuracy of these maps. Finally, I study the measured topologies for evidence of sampling biases: errors in the measured topologies that result from having a limited number of vantage points.

This chapter has two themes: that the mapping techniques are reasonably efficient and that the resulting maps are reasonably accurate. The two are not completely orthogonal: an efficient system can collect a map with fewer measurements (so fewer measurements with error) and complete quickly enough that the network does not have time to change significantly. The principle behind these techniques is to choose only the measurements likely to contribute new information.

Establishing the correctness of the measured maps is a difficult task because the true maps are not available to me for comparison. To provide confidence that the measured maps are reasonably accurate, I present the results of several comparisons. Operators from three ISPs of the ten measured checked a sample of the maps by hand and confirmed their accuracy. Separately, I estimate the completeness of the maps both by scanning ISP IP address ranges for routers that might have been missed and by comparing the peering links I find with BGP adjacencies in Route Views. A comparison with excerpts of the topology measured by Skitter [33] shows that the ISP network topologies I measure are more complete; I find roughly seven times more routers and links. A comparison with the IP addresses found by scanning suggests that the maps I uncovered are within a factor of two of being complete. Tests for sampling bias, incompleteness to the topology that varies with distance from vantage points, show that some incompleteness may remain.

I applied Rocketfuel techniques to map ten diverse ISPs during December, 2001 and January, 2002. I used 294 public traceroute servers listed by the http://traceroute.org/ Web page [68], representing 784 vantage points across the world. One traceroute server can be configured to generate traceroutes from many routers in the same autonomous system. For example, `oxide.sprintlink.net` can generate traceroutes from 30 different vantage points. The majority (277) of the public traceroute servers, however, support only one source. The BGP tables that provide input to directed probing are taken from Route Views [80]. I use a second, smaller, data set to more comprehensively evaluate the accuracy of alias resolution methods; this dataset is described in the section on alias resolution.

The results I measure are limited to the ISPs I studied, the vantage points I used, and the BGP tables I used as input. With a larger set of vantage points or BGP tables, more accurate results are likely. Different ISPs may be more or less suitable to being mapped using these techniques than the ten I selected.

5.1 The ISPs

To explore the effectiveness of Rocketfuel in mapping different types of ISPs, I chose to map a diverse group: on different continents, large and small, dense and sparse. The ten ISPs I studied are a mix specifically chosen to test the applicability of the techniques; they were not chosen to be representative of all ISPs.

Table 5.1 shows aggregate statistics about the ten mapped ISPs, highlighting their diversity. The biggest networks, AT&T, Sprint, and Verio, are up to 100 times larger than the smallest networks studied. I mapped these ISPs in December 2001 and January 2002. Since then, Ebone has stopped operation (July 2002 [18]),

Table 5.1: The number of measured routers, links, and POPs in December 2001–January 2002 for the ten ISPs studied. ISP routers include backbone and access routers. The column labeled "with customer & peer" adds directly connected customer access and peer routers. Links include only interconnections between these sets of routers. POPs are the distinct location tags found in names of routers conforming to the naming convention of the ISP.

AS	Name	ISP		with customer & peer		POPs
		Routers	Links	Routers	Links	
1221	Telstra (Australia)	377	781	2,998	3,178	61
1239	Sprintlink (US)	699	2,273	8,351	9,916	44
1755	Ebone (Europe)	171	381	608	543	26
2914	Verio (US)	1,012	2,835	7,108	7,614	122
3257	Tiscali (Europe)	242	311	788	664	51
3356	Level3 (US)	610	3,742	2,935	7,126	53
3967	Exodus (US)	214	443	916	849	24
4755	VSNL (India)	4	1	82	39	11
6461	Abovenet (US)	356	861	1,765	1,709	22
7018	AT&T (US)	655	2,078	10,151	11,680	109
	Total	4,340	13,706	35,702	43,318	523

Tiscali has merged infrastructures with acquired European ISPs [11], and Exodus declared bankruptcy and was acquired by Cable and Wireless (February 2002 [27]) which declared bankruptcy and was acquired by Savvis (January 2004 [99]).

Unlike many network mapping studies, I focus on commercial networks and excluded research networks like the Abilene (Internet2) network in the United States and DANTE in Europe. Research networks are unlikely to be representative, but are useful for other network measurement studies because many publish the correct topologies that can be used for validation [55, 59]. Unfortunately, they are unsuitable for the techniques I develop that target typical ISPs because many have restrictive routing policies that allow traffic only between participants. Specifically, these routing policies subvert the BGP directed probing techniques in Section 4.1.1: directed probing relies on some prefixes being dependent upon an ISP for connectivity and on the visibility of an ISP's routes in BGP. Directed probing techniques could be developed for the special case of research networks, and would likely be quite successful. I defer developing techniques to measure such atypical ISP networks to future work.

Section 7.1 presents the resulting maps and Section 7.2 presents an analysis of their properties.

5.2 Efficiency of Traceroute Collection

In this section, I evaluate the effectiveness of the techniques that make ISP mapping efficient by removing redundant measurements. That the accuracy of the result is still adequate is evaluated in the following section; here I am primarily concerned with how well each technique identifies and discards redundant measurements before they are taken. Each technique demonstrates its value by usually eliminating more than 90% of the possible measurements it sees. Depending on the ISP being mapped, the composition of techniques eliminates all but 0.01% to 0.3% of the all-to-all measurements.

The quantitative results I present in this section depend on the effectiveness of the techniques, but also on factors specific to this evaluation, including the source of BGP information, the number and diversity of vantage points, and the selection of ISPs being studied. Although the results are encouraging, the efficiency

Table 5.2: The effectiveness of directed probing for different ISPs. Rocketfuel executes the traceroutes chosen after path reductions filter the directed probes and executes the egress discovery traceroutes from a locally-controlled machine not subject to the same resource constraints. The total all-to-all traces includes traces to all BGP advertised prefixes, just once, and traces to the broken-down prefixes of the ten ISPs. It is not the sum of the traces because many of those will be redundant.

ISP	All-to-all	Directed Probes		Traceroutes after reduction	Egress Discov.	Pct. Remain
Telstra (Aus)	105 M	1.5 M	(1.4%)	20 K	20 K	0.04%
Sprintlink (US)	132 M	10.3 M	(7.8%)	144 K	54 K	0.15%
Ebone (Europe)	91 M	15.3 M	(16.8%)	16 K	1 K	0.02%
Verio (US)	118 M	1.6 M	(1.3%)	241 K	36 K	0.23%
Tiscali (Europe)	92 M	0.2 M	(0.2%)	6 K	2 K	0.01%
Level3 (US)	98 M	5.0 M	(5.1%)	305 K	10 K	0.32%
Exodus (US)	91 M	1.2 M	(1.3%)	24 K	1 K	0.03%
VSNL (India)	92 M	0.5 M	(0.5%)	5 K	2 K	0.01%
Abovenet (US)	92 M	0.7 M	(0.7%)	111 K	3 K	0.12%
AT&T (US)	152 M	4.5 M	(2.9%)	150 K	80 K	0.15%
Total	297 M	40.8 M	(13.7%)	1022 K	209 K	0.38%

provided by these techniques in studying these ISPs should not be construed as necessarily representative of how the techniques would perform on an arbitrary ISP. This evaluation also does not show what I expect to be true: a larger source of BGP information having more complete information would better guide the search for redundant measurements, and an even larger platform of vantage points would be well-supported by the techniques I evaluate here.

5.2.1 Directed Probing

I consider two aspects of directed probing: how much work it discards and how accurately it predicts routes. The prediction accuracy of the heuristic is distinct from the accuracy of the result, which is evaluated in the next section. I measure the effectiveness of directed probing by the fraction of traces it can prune. I measure the accuracy of its predictions in two ways: i) by the number of pruned traces that would have transited the ISP and should have been kept because they might have contributed information, and ii) the number of collected traces that should have been discarded because they did not transit the ISP. These two errors could be thought of as false positives and negatives; I avoid these terms because the positive response is ambiguous—yes, this traceroute should be skipped, or, yes, this traceroute should traverse the ISP.

Table 5.2 shows the number of traceroutes selected by directed probing. The all-to-all mapping strategy of sending traceroutes from all 784 vantage points to all 115,335 BGP-advertised prefixes, breaking larger ISP prefixes into /24s, would require 90–150 million traceroutes for the ten ISPs I studied.[1] Directed probing chooses between 0.2–17% of these traces.

The Skitter data set provides a means to estimate how many traces are incorrectly pruned by directed probing. Skitter, described in Section 3.1.2, traces from relatively few vantage points to many more destinations and does not focus on any ISP at a time. The specific destination list depends on the Skitter vantage point; in

[1] ISPs that own more, larger address space, such as AT&T's 12.0.0.0/8, require many traceroutes when large prefixes they originate are broken down to /24s.

Table 5.3: The percentage of collected traceroutes, chosen by each directed probing method, that traversed each ISP. The percentage listed is the fraction of collected traces that traversed at least one IP address owned by the ISP being mapped. There were no traceroute servers inside VSNL, so it had no insider traces.

ASN	ISP Name	Directed probing type			Overall
		Dependent Prefix	Insider	Up/down	
1221	Telstra	95.5%	97.6%	84.1%	95.5%
1239	Sprint	98.0%	99.8%	83.5%	97.0%
1755	Ebone	96.3%	94.7%	77.4%	91.4%
2914	Verio	98.6%	47.5%	92.1%	93.5%
3257	Tiscali	97.2%	99.7%	91.8%	96.8%
3356	Level3	97.4%	81.3%	82.9%	85.2%
3967	Exodus	95.8%	97.5%	89.6%	95.0%
4755	VSNL	91.4%	—	86.8%	90.9%
6461	AboveNet	94.6%	87.1%	73.6%	87.1%
7018	AT&T	98.9%	97.3%	92.2%	97.7%
Overall:		97.9%	83.2%	85.6%	93.8%

December 2001, most of the 15 vantage points used a destination list of 58,224 addresses and took 4.8 million traceroutes (iterating over the destination list several times). Skitter traceroutes provide the means to answer the question: how many useful traceroutes would directed probing have mistakenly told Skitter to skip? From 0.1% to 7% of all the traceroutes that traverse each ISP were useful but pruned by directed probing. This percentage is low for non-US ISPs like VSNL (4755) and Tiscali (3257), and higher for the big US ISPs like AT&T and Sprint. The variation by ISP may be due to the difference in the likelihood that a trace to a randomly selected destination will traverse the ISP. Big US ISPs have more useful but pruned traces because a plurality of IP addresses are located in the US. I did not explore how many of these potentially useful traces would traverse new paths: subsequent path reductions may find that pruned traces are redundant with others. The potentially missing data does not seem to compromise the completeness of the maps significantly, as I will show in Section 5.6.

To determine how many collected traceroutes were unnecessary, I tally directly from the measurements: 6% of the traceroutes collected to map these ten ISPs did not transit the intended ISP and were not necessary. Table 5.3 presents the fraction of traceroutes selected by each of the directed probing methods on each of the ten ISPs I studied. Notable is the low prediction accuracy for Verio insiders: only 47.5% of traceroutes from the two vantage points thought to be within Verio traversed Verio's network. Both of these vantage points had address space originated by Verio, but also had connectivity from other ISPs—this problem could have been detected on-line after a few tens of traceroutes and the vantage point reclassified as not an insider. Also notable is the relatively low prediction accuracy of up/down traceroutes for many ISPs. Recent studies have found comparable disagreement between the AS-paths in BGP and actual forwarding paths taken [5, 60, 75], even when both are observed at the same location, suggesting that these prediction errors may be inherent in the approach of guiding traceroute with BGP routing information.

These results are encouraging: not only does directed probing eliminate the large majority (83–99.7%) of unnecessary traceroutes, but little useful work is pruned out (0.1–7%), and little useless work is done (2.3–14.8%).

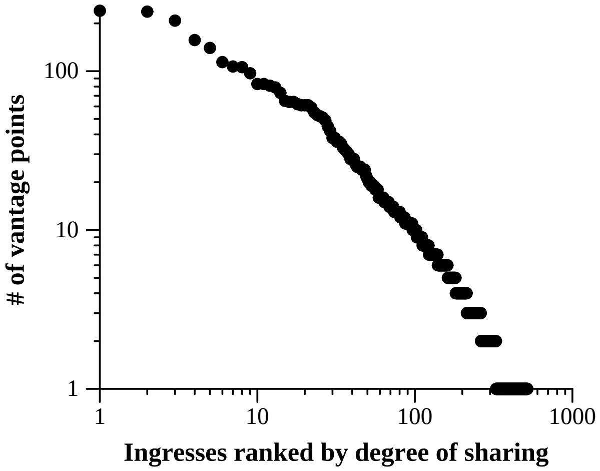


Figure 5.1: The number of vantage points that share an ingress, by rank, aggregated across ISPs. Each x-axis value represents an ingress. Ingresses on the left are widely shared by many vantage points: at the left edge of the graph, 232 vantage points share the same ingress into one ISP. Ingresses on the right are shared by very few: 247 vantage points have unique ingresses. The vantage points found 517 ingresses across all ISPs.

5.2.2 Path Reductions

Path reductions choose among traceroutes likely to traverse the ISP, which traces are likely to discover new paths. In this subsection, I evaluate ingress, egress, and next-hop AS path reductions independently and in combination. Recall that directed probing first eliminates traceroutes unlikely to traverse the ISP; path reductions eliminate traceroutes that are likely to provide redundant information. Refer to Section 4.1.2 for a description of how path reductions work.

5.2.2.1 Ingress reduction

Ingress reduction groups vantage points when they share an ingress to the mapped ISP, avoiding redundant measurements and allowing the vantage points to share the rest. Ingress reduction kept 2–26% (12% overall) of the traces chosen by directed probing. For VSNL, ingress reduction kept only 2% of the traces because there were only six ingresses for the vantage points. In contrast, it kept 26% of the traces chosen by directed probing of Sprint: the larger ISP had 75 unique ingresses for the vantage points used in this measurement.

To explain this variation and the efficiency gained through ingress reduction, I study the sharing of ingresses by vantage points. Figure 5.1 shows the distribution of vantage points that share an ingress. I sort the ingresses in decreasing order of the number of vantage points share that ingress. I combined all ingresses from all ISPs in this plot, and show the result on a log-log scale. The right side of the curve shows that fidelity would be lost if vantage points were removed: many provide distinct ingresses into a mapped ISP. At the left, many vantage points share only a few ingresses. These vantage points can share their work, allowing ingress reduction to reduce the amount of work necessary, even after directed probing. The leftmost point in the graph represents 232 vantage points that shared an ingress into Ebone, a European ISP, in London. At right, 189 ingresses across the ten mapped ISPs were not shared: the vantage points using these ingresses contributed new information to the map.

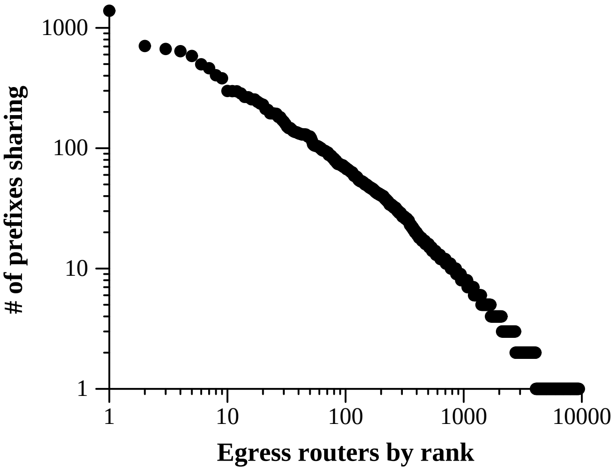


Figure 5.2: The number of dependent prefixes that share each egress router, ranked in decreasing order, and aggregated across the ten ISPs. As in Figure 5.1, at left, many prefixes share a few egresses: 1,387 prefixes share an egress in Telstra. At right, many prefixes have a single egress router, showing the need for many unique destinations.

Extra vantage points contribute either speed or accuracy. Speed is increased when the new vantage point shares an ingress with an existing vantage point because more traceroutes can execute in parallel. Accuracy is improved if the new vantage point has a unique ingress to the ISP.

5.2.2.2 Egress reduction

Egress reduction groups destination prefixes that share an egress router, avoiding redundant measurements to the individual destinations beyond an egress router. Egress reduction kept only 13% of the dependent-prefix traceroutes chosen by directed probing. Recall that dependent-prefix traceroutes are those to destinations that appear to depend on the mapped ISP for connectivity. Because egress reduction is symmetric to ingress reduction which groups vantage points that share an ingress, the explanation of the effectiveness of egress reduction is similar.

Table 5.4 shows the number of distinct egress router addresses discovered when tracing to the dependent prefixes of each ISP. Egress router addresses are used because this step of mapping the ISP precedes alias resolution. The ratio of egress router addresses to dependent prefixes represents the effectiveness of the reduction: only 4–22% of the traceroutes provide new information about the topology. The variation in the effectiveness of egress reduction appears not to be caused solely by the size of the ISP: the technique is least effective on the largest ISPs (AT&T and Sprint) and the smallest ISP (VSNL). The differences may be related to how ISPs allocate and aggregate customer prefixes, the size of customer networks, and the number of access routers used.

Figure 5.2 shows the number of dependent prefixes that share an egress router. I present the specific procedure for discovering egress routers in Section 4.1.3. The x-axis represents each egress router and the y-axis value represents how many dependent prefixes share that egress.[2] I combined all egress routers from all ISPs in this plot, and show the result on a log-log scale. The left part of the curve depicts egresses shared by several prefixes, where egress reduction is effective. The largest egress router was one in Melbourne in Telstra, having

[2] Unlike vantage points that each elect an ingress based on the majority of observed ingresses in traceroutes, dependent prefixes are bound permanently to egress routers in egress discovery. As a result, dependent prefixes that share an egress *always* share that egress.

Table 5.4: The number of observed egress router IP addresses and dependent prefixes for each of the ten mapped ISPs. The percentage represents the fraction of dependent-prefix traceroutes needed per vantage point (or ingress after ingress reduction).

ISP (ASN)	Dependent Prefixes	Egress Router Addresses	Percentage
Telstra (1221)	8,962	914	10.2%
Sprint (1239)	16,325	3,322	20.3%
Ebone (1755)	7,036	274	3.9%
Verio (2914)	12,484	1,569	12.6%
Tiscali (3257)	2,501	139	5.6%
Level3 (3356)	10,410	1,006	9.7%
Exodus (3967)	4,315	404	9.4%
VSNL (4755)	1,261	180	14.3%
Abovenet (6461)	9,612	465	4.8%
AT&T (7018)	18,900	4,086	21.6%
Overall	91,806	12,359	13.5%

1,387 dependent /24 prefixes. The right part shows that 5,353 prefixes had unique egresses. Traceroutes to these prefixes are likely to add detail to the map.

Like ingress reduction, egress reduction preserves detail while limiting redundant measurement.

To test my hypothesis that breaking larger prefixes into /24s is sufficient for discovering egresses, I randomly chose 100 dependent /24 prefixes (50 owned by the ISP; 50 by customers of the ISP) and broke them down further into /30 prefixes. I then traced to a randomly chosen address in each /30 of these /24s from a local machine. The ratio of egresses discovered when tracing to /30s to the total egresses discovered overall is an estimate of lost accuracy. Depending on the ISP, 0–20% of the egresses discovered during this process were previously unseen, with the median at 8%. This wide range suggests that breaking prefixes to /24s, while sufficient for some ISPs (two had almost no new egresses), is not universally applicable: some ISPs allocate prefixes smaller than a /24 to customers.

5.2.2.3 Next-Hop AS reduction

Next-hop reduction groups prefixes that share the same next-hop AS, avoiding redundant measurement to the individual prefixes that are likely to be reached through the same peering link. Next-hop AS reduction selected 5% of the up/down and insider traces chosen by directed probing for the ISPs I studied and vantage points I used. Recall that these two classes leave the ISP and proceed to enter another AS: the next hop. Next-hop AS reduction groups destinations likely to leave the mapped ISP at the same peering link.

In Table 5.5, I show the effectiveness of next-hop AS reduction for each ISP. Unlike egress reduction, which applies equally to all vantage points because it affects the dependent prefixes, next-hop AS reduction applies differently to some vantage points: insiders and those often in Route Views paths selected for up/down traces. So, while Table 5.4 compares the much smaller number of prefixes and router addresses, Table 5.5 shows the overall number of traceroutes chosen. Next-hop AS reduction appears particularly effective for Ebone. Ebone had 127 insider vantage points; without the reduction, each of these vantage points would have had 116,211 traceroutes to complete.

Table 5.5: The effectiveness of next-hop AS reduction at removing up/down and insider traceroutes selected by directed probing. The percentage represents the fraction of up/down and insider traceroutes needed overall.

ISP (ASN)	Up/down and Insider Traceroutes	After Next-hop Reduction	Percentage
Telstra (1221)	473,657	54,732	11.6%
Sprint (1239)	9,743,892	665,275	6.8%
Ebone (1755)	15,320,901	167,694	1.1%
Verio (2914)	1,307,462	173,761	13.3%
Tiscali (3257)	181,515	12,825	7.1%
Level3 (3356)	4,890,789	124,453	2.5%
Exodus (3967)	577,420	93,167	16.1%
VSNL (4755)	266,528	24,498	9.2%
Abovenet (6461)	693,503	84,220	12.1%
AT&T (7018)	3,594,007	432,745	12.0%
Overall	37,049,674	1,833,370	5.0%

To help explain this effectiveness, Figure 5.3 shows, for each vantage point along the x-axis, the number of destination prefixes chosen by directed probing (the upper markers) and the number of next-hop ASes that represent traceroutes after reduction (the lower markers). Next-hop reduction is effective because the number of next-hop ASes is much smaller than the number of prefixes. It is particularly valuable for reducing the workload of vantage points inside ISPs. Without next-hop AS reduction, these insider vantage points would traceroute to all of the prefixes in the Route Views BGP table. Next-hop AS reduction allows insiders to trace instead to many fewer external destinations: only those needed to cover the set of possible next-hop ASes.

Next-hop AS reduction achieves this savings by assuming that routes are chosen based solely on the next-hop AS and not differently for each prefix advertised by the downstream ISP. Commonly, route selection by next-hop AS and not destination prefix is equivalent to whether “early exit” routing is used between these ISPs.[3] However, the reduction preserves completeness as long as the traces from each ingress to randomly-chosen prefixes in the next-hop AS find each link to that AS. I estimate how frequently this assumption is violated by conducting the 600,000 traces that would be needed to map Verio without the reduction. The traces contained 2500 (ingress, next-hop AS) pairs, of which only 7% violated the assumption by including more than one egress. Different ISPs have different policies regarding per-prefix inter-domain routing, but nevertheless this result is encouraging.

It remains future work to apply an understanding of peering routing policy, as developed in Chapter 6, to determine whether Next-hop AS reduction is appropriate, and if not, to choose geographically diverse prefixes as destinations to discover as many peering points as possible.

5.2.2.4 Overall efficiency of path reductions

Ingress and egress/next-hop AS reductions are orthogonal and compose to give multiplicative benefit. Table 5.2 shows the number of traceroutes I collected to infer the maps. Overall, Rocketfuel techniques chose less than

[3] Quantifying how often early-exit routing is used between pairs of ISPs is difficult because there are many small ISPs; small ISPs with only one connection always use early-exit because there is no alternative. Although between the largest ISPs, as I will show in Chapter 7, early-exit seems to be used about half the time, the rest can be classified into engineered (mostly early-exit) or late-exit, neither of which is troublesome for next-hop AS reduction (see 4.1.2).

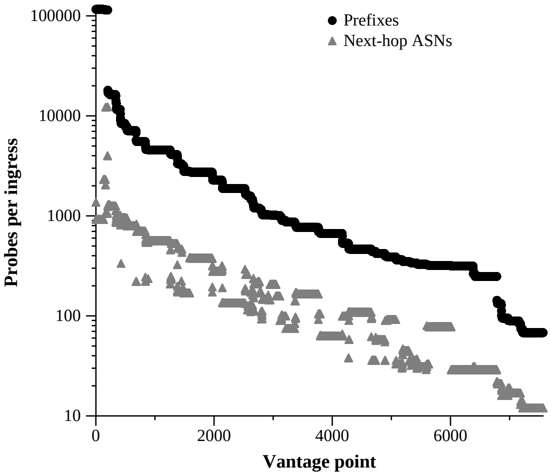


Figure 5.3: The number of prefixes chosen by directed probing and unique next-hop ASes for each vantage point, sorted in decreasing order by prefixes. A vantage point is counted once for each mapped ISP. The leftmost point is an insider mapping AboveNet, choosing 116,277 prefixes, which is then reduced to 1,327 next-hop ASes. The most work required is for an insider in Telstra, which is only reduced to 12,317 from 114,333 destination prefixes. (The number of destinations varies because some prefixes are inside the ISP.)

0.1% of the traces required by an all-to-all technique. Depending on the ISP, the number of traces after reduction varied between 0.3% (Level3) to 0.01% (VSNL and Tiscali).

Rocketfuel mapping techniques also scale with the number of vantage points. Extra vantage points contribute either speed or accuracy. Speed is increased when the new vantage point shares an ingress with an existing vantage point because more traceroutes can execute in parallel. Accuracy is improved if the new vantage point has a unique ingress to the ISP.

5.3 Agreement Between Alias Resolution Methods

An alias resolution technique is accurate when its statements about whether IP addresses represent the same router are correct. In this section, I first evaluate how well false positives are removed when using the IP-identifier technique. I then compare the IP-identifier technique with the source-address based technique used by Mercator in earlier work and a DNS-based alias resolution tool that uses undns, described in Section 4.3.3. Because true ISP maps are not available to me, comparisons between the results of individual methods represent the limit of my ability to evaluate the accuracy of each method. Finally, I estimate the completeness of the techniques, used alone and in concert. Comparing measured aliases to true ISP network topologies is future work.

The evaluation of alias resolution methods was completed in May 2003 using two separate topologies: an exhaustively-mapped PlanetLab [97] overlay topology and a map of a large ISP, UUnet, collected using Rocketfuel's directed probing techniques. They were collected specifically for evaluating alias resolution techniques and are not used elsewhere in this dissertation. The advantages of using this restricted data-set for evaluation of alias resolution are several. First, the smaller overlay can be exhaustively probed, ensuring that heuristics for guiding the alias test do not miss aliases. Second, the overlay topology is diverse, including the routers of 95 ISPs. This diversity is useful for evaluating alias resolution techniques because they depend on implementation idiosyncrasies of routers from different vendors; many ISPs purchase routers from only a single vendor like

Table 5.6: False alias statistics from using Ally on PlanetLab and UUnet. A first pass performs many tests. Of those few pairs that appear to be aliases, a second pass confirms or disproves the alias pair. Although many alias pairs were initially believed but disproven in the second pass, the error rate of the test itself is very low, indicating that a single verification pass is sufficient.

		Alias pairs		Test false
Map	Tests performed	Confirmed	Disproven	positive rate
PlanetLab	265 million	557	120 (22%)	5×10^{-7}
UUnet	0.2 million	2,782	64 (2%)	3×10^{-4}

Cisco or Juniper. A technique that is effective on one ISP may not work for others, so having many different ISPs in the dataset is useful. Third, the UUnet ISP is considered the largest and I use it to show the scalability of alias resolution for ISP topology mapping.

Some inaccuracy in the Rocketfuel maps was caused by inaccurate alias resolution, but it was difficult to tell exactly why errors, or at least inconsistencies, occurred. In part, the difficulty of evaluating alias resolution is that DNS names are the best indication available for what ISPs believe are the true aliases, yet some DNS names are out-of-date or mistyped. The evaluation in this section is an attempt to understand the problem of alias resolution independently, to treat both probing methods and DNS-based methods on an equal footing.

Two IP addresses are *aliases*, as described in Section 4.2, if they represent interfaces on the same router. In this section, I will focus quantitatively on *alias-pairs*, or the number of pairwise aliases that are found. A router with two aliases will have one alias-pair; three aliases, three alias-pairs; four aliases, 6 alias pairs. Not all of these alias-pairs must be discovered directly, but this metric more closely matches the pairwise alias test being evaluated. Most routers have only a few discovered aliases, so the results are not biased by routers that have many aliases.

5.3.1 IP Identifier False-Positives

The IP identifier technique infers the sharing of a single counter between two interfaces that are aliases. By random chance, distinct counters may be temporarily synchronized and appear to be a single counter. The purpose of this section is to show that a single verification phase is both necessary and sufficient for removing false aliases detected by the IP identifier test. Verification is necessary because the false positive rate, relative to the size of the topology, is significant enough to cause inaccuracy in the map. A single verification phase is sufficient because the false positive rate is quite small relative to the number of aliases discovered. Confusion about the accuracy of the pairwise test has fostered derivative approaches that detect aliases when the test returns positive 80 of 100 times [42], which is careful and safe, but, as I will show, unnecessary.

To determine the false positive rate of the IP-identifier alias resolution test, I count how many alias pairs were confirmed by a verification phase on both the PlanetLab and UUnet topologies. To support the evaluation of heuristics to guide alias resolution, described in Section 5.4, I tested all pairs of addresses in the PlanetLab topology. I tested a subset of the address pairs in the UUnet topology, guided by heuristics that choose address pairs likely to be aliases. Table 5.6 shows the results of the verification phase. The pairs that the verification phase again classifies as aliases are considered “confirmed” while the rest are “disproven.” These disproven aliases represent inaccuracy removed from the resulting map by the verification phase.

In Table 5.6, I show the false positive rate relative both to the “confirmed” aliases and to the total number of tests performed. Despite the small size of the PlanetLab topology, 265 million alias-pairs are tested, many more than would be necessary if TTL replies were used to filter candidates. The 120 pairs that were falsely believed to be aliases yield an error rate of 1 in 2 million tests. Fewer pairs were tested over the topology of UUnet,

Table 5.7: PlanetLab: Error rate Of alias resolution techniques. I compare aliases found using each method to those found by Ally or DNS. In each column is the number of disagreements over the number of pairs of addresses about which both techniques made a statement: this baseline changes because the techniques cover disjoint sets of address pairs. No discrepancies were found.

Tested	**Reference Technique**	
Technique	**Ally**	**DNS**
Mercator false positive	0/ 382 (0%)	0/ 185 (0%)
Ally false positive	–	0/ 334 (0%)
Ally false negative	–	0/12,881 (0%)
DNS false positive	0/ 334 (0%)	–
DNS false negative	0/12,881 (0%)	–

and the test showed a false positive rate of 1 in 3 thousand. The UUnet topology is more homogeneous and interdependent—it is likely that IP-identifier values are more synchronized because routers within a domain all participate in the same routing protocol. This table shows that the false positive rate inherent in the approach is very low, but not that the method has no systematic errors.

I investigated the cause of these false positives and found that the individual IP identifier test is less accurate when routers rate-limit responses. When routers rate-limit responses, it is impossible to obtain four closely-spaced replies from the router. Instead the tool looks for the proximity of IP identifiers in the packets it receives. Proximity is less convincing evidence than what the full technique provides. I use generous thresholds that favor false positives (over false negatives); the verification phase catches these false positives, allowing rate-limiting routers to be accurately mapped. Rate-limiting routers are not a major concern because they are resolvable with the source-address-based technique described in Section 4.2: the implementation decision to send traffic with the address of the outgoing interface as the source seems correlated with the implementation decision to rate-limit responses.

Although the number of false positives appears insignificant relative to the number of tests performed, it is substantial relative to the number of confirmed aliases and can lead to inaccuracy in the resulting map. A verification phase to verify the initial set of aliases and discard false positives from the set is thus essential. Fortunately, the low false-positive rate ensures that a single verification phase is sufficient to detect and discard false positives.

5.3.2 Comparative Evaluation

Next, I measure the agreement of each technique by comparing the results of each pair of techniques. Agreement between independent techniques suggests, but does not prove, the accuracy of each method. I compute the error rates as the percentage of cases in which a pair of techniques disagree on a classification of an alias or a non-alias pair. The “false positive” rate of a technique is the likelihood that its assertion that a pair of addresses are indeed aliases will be disputed by a reference technique. The “false negative” rate is the complement, or the likelihood that a pair classified as not-aliases will be disputed by a reference technique. “False positive” and “false negative” are quoted because they are taken relative to a reference technique, *not* to absolute truth, which is unavailable to me.

Tables 5.7 and 5.8 summarize the error rates of each technique for the mapping of PlanetLab and UUnet. For PlanetLab, Mercator, Ally, and DNS show zero false positives and negatives when compared to the other techniques. (I do not quantify “false negatives” with Mercator because the technique does not disprove aliases.) For UUnet, the alias pairs that represent false positives for Mercator and Ally relative to DNS are consistent—

Table 5.8: UUnet: Error Rate of alias resolution techniques. I compare aliases found using each method to those found by Ally or DNS. In each column is the number of disagreements over the number of pairs of addresses about which both techniques made a statement: this baseline changes because the techniques cover disjoint sets of address pairs.

Tested	**Reference Technique**	
Technique	**Ally**	**DNS**
Mercator false positive	3/ 1,293 (0.2%)	23/ 410 (5.6%)
Ally false positive	–	11/ 965 (1.1%)
Ally false negative	–	6/17,633 (0.03%)
DNS false positive	6/ 960 (0.6%)	–
DNS false negative	11/17,638 (0.06%)	–

both tools assert that some addresses are aliases while DNS disagrees, suggesting incorrect or simply out of date DNS names.

5.3.3 Completeness

I define "completeness" in alias resolution to be the fraction of aliases discovered by a technique to the total number of aliases among the addresses in the map. Because no true map is available to show how many aliases are in the network, I consider the union of aliases identified by all techniques as the total. I then compare the number of aliases discovered by a single technique or combination of techniques to this total. In all likelihood, the union of discovered aliases is not the total number of true aliases in the network: each estimate of completeness is likely to be an overestimate, though it may be an underestimate if the false positive rate of any technique is high. In this section, I first study the improvement offered by multiple vantage points to source-address based alias resolution, and then compare the "completeness" of the different techniques.

Each technique can contribute different groups of aliases, so the overall completeness of alias resolution is improved when they are used in concert. (The combination of single-vantage-point Mercator and the three-packet Ally test were used to form the maps in the rest of this chapter.) The IP addresses discovered by traceroute may be unsuitable for the alias resolution techniques because they use private address space [107], unroutable address space, are unresponsive to further probing, or, worse, are anonymous in that they do not respond even to initial traceroute probes [134]. These problems are rare in my experience with the ISPs I studied, but can prevent an accurate topology from being assembled. Techniques like Mercator's and Ally's require that the router be *responsive*: that it send replies to probes directed at its interfaces from at least one vantage point. The DNS techniques have no such requirement. Different implementation decisions in routers influence the effectiveness of Mercator's and Ally's techniques.

Additional vantage points increase the completeness of the source-address-based alias resolution method. Figure 5.4 shows the aliases found when using the source-address-based method from up to eleven vantage points. While the first vantage point discovered 655 alias pairs, using all eleven vantage points found 1,271, nearly doubling the number of discovered aliases. (The number discovered by a single vantage point varied between 539 to 663.) This shows that more vantage points help. These extra vantage points serve the same purpose as source-routed probes in Mercator: some router addresses can only be reached by certain vantage points. For UUnet, eleven vantage points appear to be enough to gain most of the benefit. For PlanetLab, this effect was less pronounced; a greater fraction of the source-address aliases was found by the first vantage point.

Tables 5.9 and 5.10 show the "completeness" of each alias resolution approach for PlanetLab and UUnet. Although individual approaches find at most 80% of the aliases in the network, using them in combination is

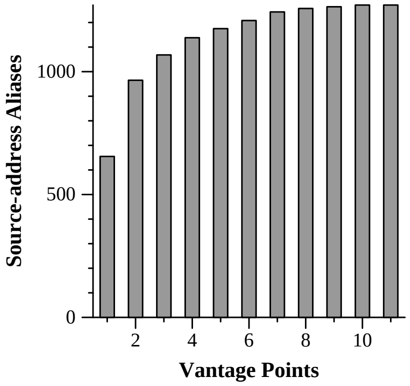


Figure 5.4: Each vantage point contributes to a source-address based alias-resolution technique in the mapping of UUnet. The order of the vantage points is chosen randomly and appears not to matter.

Table 5.9: PlanetLab: "Completeness" of techniques. I define the union of aliases found by the three techniques to be 100%.

Technique group	**Of 727 alias-pairs**	**Of 654 responsive alias-pairs**
Mercator	345 (47%)	345 (53%)
Ally	557 (77%)	557 (85%)
DNS	331 (46%)	258 (39%)
Mercator ∪ Ally	608 (84%)	608 (93%)
Mercator ∪ Ally ∪ DNS	727 (100%)	654 (100%)

beneficial. Using DNS names as an alias resolution method can find aliases Ally and Mercator cannot. Some of these extra aliases are unresponsive routers. Most of the aliases from responsive routers in PlanetLab that were missed by Ally were caused by addresses that were only temporarily responsive: these did not respond when probed by Ally. Nearly all routers in UUnet were responsive, so there is very little difference between the completeness of techniques when considering only responsive addresses, so only one column of completeness results is in Table 5.10 for UUnet.

5.4 Efficiency of Alias Resolution

The efficiency of the source-address- and DNS-based methods is clear: each use only a few packets per address.[4] The topic of this section is how to make a pair-wise alias resolution test, in particular the IP-identifier test, efficient. Without heuristics to guide the search for alias pairs, every address would be tested against every other address: quickly becoming unusable for even moderately-sized networks. Using the PlanetLab dataset described above, I quantified the importance of heuristics that guide the search for alias pairs. These heuristics

[4] DNS lookups take only a few packets to traverse the hierarchy. Source-address tests may be run from several vantage points, but each vantage point sends only one packet, retransmitting as needed.

Table 5.10: UUnet: "Completeness" of techniques. Nearly all routers were responsive (3,378 of 3,421), so a second column is not shown.

Technique group	Of 3,421 Overall
Mercator	1,271 (37.2%)
Ally	2,782 (81.3%)
DNS	1,290 (37.7%)
Mercator ∪ Ally	3,086 (90.2%)
Mercator ∪ Ally ∪ DNS	3,421 (100%)

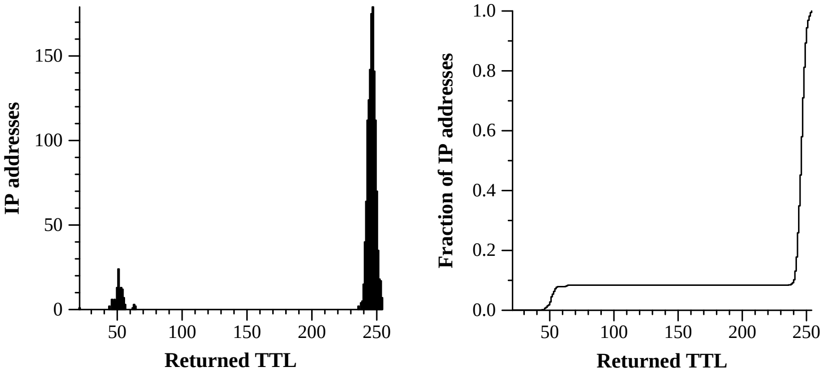


Figure 5.5: Distribution of return TTLs as seen from my site to the addresses in the PlanetLab dataset. The two modes represent different implementation decisions: some routers use an initial TTL of 255 and others use an initial TTL of 64. Both graphs depict the same data—at left is a histogram, at right, a cumulative distribution function (CDF).

include comparing only within clusters formed by candidate aliases sharing the same return TTL and sorting by piecewise reversed DNS name.

Responses that have in-sequence IP identifiers from two different IP addresses provide evidence that the addresses are aliases. To solicit these responses, pairs of interface addresses must be probed individually, and this process can require many packets. The problem addressed in this section is how to choose pairs of addresses that should be tested so that the process balances efficiency, choosing few pairs, and completeness, choosing the pairs that are likely to be aliases.

To better guide the search for aliases, I apply the two heuristics described in Section 4.2.2. First, I test only the addresses for which responses have similar return TTLs—addresses that have paths of the same length (in hops) back to the measurement source. In this section, I show that the return TTL can be used to prune the search space relative to the naive approach without missing legitimate alias pairs. My second heuristic is to test addresses having similar names first, relying on the implicit structure of DNS names to identify most aliases quickly.

Recall from Section 5.3 that the evaluation of alias resolution methods in this chapter uses the measured PlanetLab and UUnet topologies collected in May, 2003. The IP identifier test used the four-packet method described in Section 4.2.

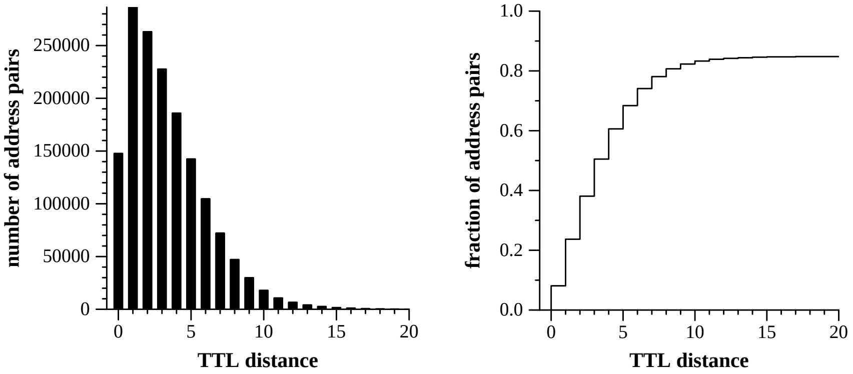


Figure 5.6: Distribution of the distance between return TTLs seen from my site for addresses in the PlanetLab dataset. Fewer than 10% of address-pairs share the same return TTL, but the median distance is only 3 hops.

The goal of the TTL heuristic is to classify addresses into groups by their location in the network, so that addresses in different locations need not be tested. The TTL heuristic thus “disproves” all candidate alias pairs when the addresses lie in different clusters. In Figure 5.5, I show the distribution of return TTLs as seen by my measurement host. The return TTL is the value in the outer IP header of the ICMP error message sent by the router. This return TTL is distinct from the outgoing TTL in the packet header encapsulated by the ICMP error message, which is inherently less accurate because routes toward aliases are less likely to be consistent than those from aliases. (Consider the two routes from a wired machine to the laptop having both wired and wireless interfaces in Figure 2.3 on page 10: the direct path takes one hop, while a second hop is needed to reach the wireless interface not on the local subnet.)

In Figure 5.6, I show the distribution of differences in return TTL between addresses in the PlanetLab dataset as seen from my machine. Address pairs that share the same TTL have distance 0, and if, for example, one’s response has a TTL of 250 and the other’s response has 251, they have a distance of 1. From the CDF, observe that testing only pairs with matching TTLs requires fewer than 10% of the all-pairs alias probes. However, I found one alias pair with a distance of 1; to catch this alias with the others would require 25% of the all-to-all probes for the PlanetLab dataset. Such small changes are expected if routing changes that affect path length occur while return TTLs are being collected.

By adding more measurement points from which to capture the return TTL, more candidate pairs can be eliminated while allowing for the rare case of aliases at different distances. Using a two-dimensional geometric metaphor, a single vantage point having the return TTL of a router can place the router somewhere on a “circle” at that distance. A second vantage point in another location can place the router at one of the two places where the circles for each vantage point overlap. Instead of using circles, I treat the return TTLs as a coordinates in an n-dimensional space, where n is the number of vantage points, and test alias pairs nearby in this coordinate space.

In Figure 5.7, I show the cumulative fraction of address pairs with increasing Euclidean distance for one to five vantage points. I chose the vantage points for geographic diversity and reliability, including PlanetLab machines at the University of Washington, Princeton University, Rice University, Datalogisk Institut Copenhagen, and the Chinese University of Hong Kong. I compute the Euclidean distance by giving each address a coordinate in n-dimensional space based on the return TTLs from each vantage point, then calculate the square root of the sum of the squares of the differences.[5] (In practice, some return TTLs will not be available; the

[5] I considered alternative methods, including minimizing the absolute (Manhattan) distance or otherwise penalizing address pairs

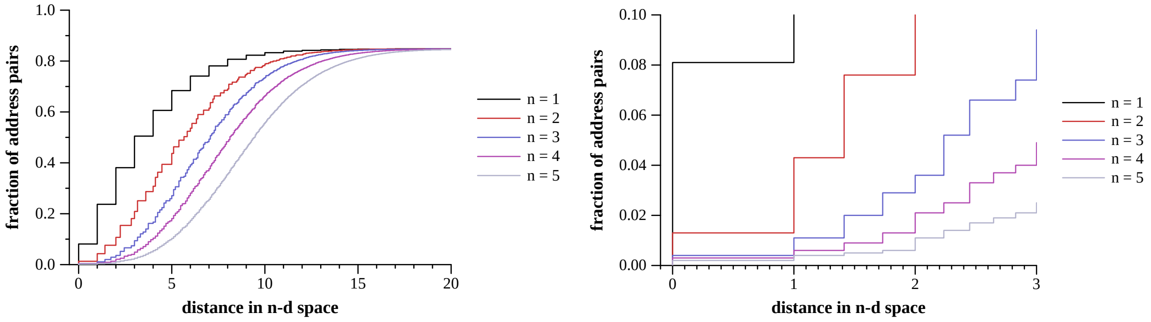


Figure 5.7: The distribution of the Euclidean distance between return TTLs when measured from one to five vantage points. The lower graph shows the lower left corner of the upper. Collecting return TTLs from more vantage points (increasing n from 1 to 5) reduces the number of address pairs that must be tested for aliasing, while still allowing some tolerance for error in the TTL measurement.

distance between addresses is taken over the return TTL values that are defined for both addresses.)

Additional vantage points permit some tolerance in the returned TTL values without sacrificing efficiency. In my tests over the PlanetLab topology, I found a single alias pair at distance three; to catch that alias would only require 2% of all-pairs alias probes with all five chosen vantage points, but 50% with just one. (More vantage points do, however, increase the likelihood that differences will be observed.)

In Figure 5.8, I present an evaluation of the second heuristic for gaining efficiency. After performing a reverse lookup (Section 3.3) on every address, addresses with similar names can be grouped. These similar names suggest similar owner, location, and role: information that can help identify a router. I sort DNS names "piecewise reversed" to preserve the hierarchical structure of the names without directly decoding their structure. Figure 5.8 shows the aliases discovered when considering only addresses that are near each other in this sorted list. Most (75%) aliases can be found by considering only addresses with adjacent names. Because "is an alias for" is a transitive relation, testing adjacent names finds groups of consecutive names that are aliases. This can reduce the number of tests needed, but only slightly. The observation that 75% of aliases can be found without using the return-TTL heuristic, however, reduces the dependence on accurate TTL clustering.

5.5 Completeness and Consistency of Inferred Geographic Locations

In this section, I evaluate how well information about the geographic structure of the measured ISP topologies was inferred from DNS names. As described in Section 4.3.2, each router in the map is assigned a physical location through the following algorithm. First, the router name for each IP address of the router is converted into a location using `undns`. If names conflict, the location of the router is elected by the majority of the names. If the router has no names, its location is inferred from the locations of its neighbors when all neighbors are in the same location. Any router that does not match any of these rules—has no name and connects to routers known to be in different POPs—has unknown location.

I evaluate the quality of the inferred structure of the maps in two ways. The first way is its completeness: how many routers have inferred location. This evaluation is straightforward. The second is its correctness: whether the assigned locations reflect the true locations of the routers. This evaluation is more difficult. Because the true

where multiple vantage points disagreed, but chose Euclidean distance because I expected it to scale gracefully to many vantage points.

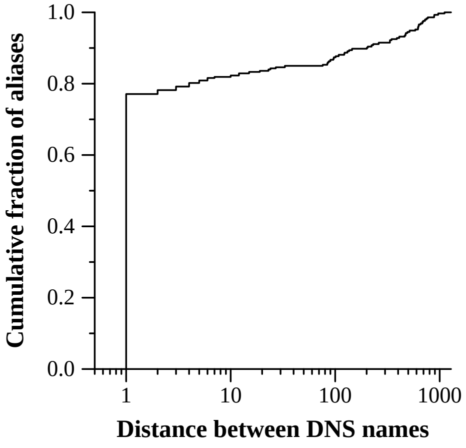


Figure 5.8: The distribution of aliases found as address pairs with increasingly different DNS names are considered. The distance between DNS names is the difference between their indices in a sorted list. Most aliases are found when testing address pairs with the most similar names; as the distance between DNS names is increased, relatively few new aliases are found. Recall that not all interfaces on a router will be discovered through traceroute. (See Figure 4.3 in which many router interfaces are not found by traceroute).

topologies are unavailable to me, I perform a weaker check that the inferred locations are consistent. Measured round-trip-times between vantage points and routers provide a means to bound the physical distance between them, and thus disprove inferred locations.

5.5.1 Completeness

Completeness in the assignment of routers to locations is the fraction of routers for which the analysis assigns a location. Some routers have no explicit location information in the names of their interface addresses. The location of these routers is inferred, when unambiguous, from the location of its neighbors. The question here is, how many routers remain in ambiguous locations, connected either to routers of differing locations or connected to other routers of unknown location?

In Table 5.11, I show the percentage of routers with unknown location and the percentage of routers having a disproven location, described in the next subsection. These routers include customer and peer routers; the total number of which can be found in Table 5.1. The ISPs having the highest percentage of unknown-location routers are Tiscali and VSNL. There were two challenges with VSNL. First, the naming convention used for routers was not as uniform as it is for the larger ISPs. Second, when names were used, it was not always clear whether a different name represented a different location, which subverts the inference of router location from the location of its neighbors.[6] Tiscali was in the process of merging with another ISP, Nacamar, and also had several, incomplete naming conventions.

The high completeness of location assignment suggests that interpreting DNS names and inferring the locations of neighboring routers is sufficient for annotating the geographic structure of an ISP. This approach works, I believe, because location information is most frequently assigned to the backbone routers, and is usually only omitted from the access routers that are connected only to backbone routers that have an assigned location. What remains is to determine whether these location annotations are correct.

[6] As one example, I remain unsure what “LVSB” in a VSNL name means, though I am told it implies that the router is in Mumbai.

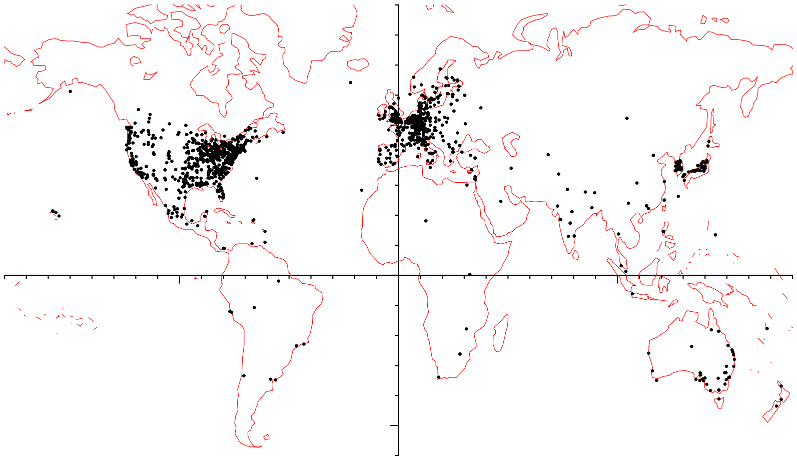


Figure 5.9: Geographic locations known to the undns library. The geographic coordinates are overlaid on an approximate map of the world. North America, Europe, and Japan have many known locations.

5.5.2 Consistency

Some challenges make it difficult to determine the correctness of inferred geographic location. First, the precision of the geographic location varies: some ISPs use names that indicate specific suburbs while others use names that indicate metropolitan areas. Second, the true locations are unavailable to me; I can only evaluate whether a router's inferred location is physically plausible. I now define the experiment to evaluate how many routers are assigned to implausible, "disproven," locations.

The putative location of a router can be disproven if it can be contacted by a vantage point in too short a time for a signal to reach the router and return. Recall from Figure 3.1 in Section 3.1 that each line of traceroute output includes both a discovered router and the time taken to receive its response, the round-trip-time. I use the physical propagation time of light in fiber[7] to determine whether the round trip time is too short for the router to be in its supposed location. For example, if a vantage point in Seattle observes a round-trip-time of 5 ms, the router cannot possibly be in Chicago, which could only be reached from Seattle with a round trip time of approximately 28 ms. The time to contact a router can be much longer—delays can be caused by queuing and processing delay or by indirect wiring paths—it simply cannot be shorter than the time for light to reach the router and return.[8]

Unfortunately, this simple strategy is not easy to apply: the locations of the vantage points are also unavailable to me. Instead, I try all possible vantage point locations. Figure 5.9 shows the locations that are possible vantage point sites. Out of all possible vantage point locations, I choose the vantage point location that is consistent with most of the observed round-trip times and proposed router locations. If some routers remain too far away geographically for the low latency with which they can be contacted from the "best fit" vantage point location, the locations of these routers are considered "disproven." The fraction of routers with "disproven"

[7] The speed of light in fiber is 2.0×10^9 meters per second, slightly slower than the speed of electrical signals in copper (2.3×10^9), which is slower than the speed of light in a vacuum.

[8] Some measurement error is possible: when clocks behave badly, they can go back in time or advance too slowly, on rare occasion causing incorrect round trip time measurements. Further, the precision of traceroute latency measurements varies: some have microsecond readings, others appear to use clocks with 4 ms precision.

Table 5.11: Coverage and plausibility of assigned locations. Of the routers in the ISP, what percentage have unknown location and what percentage have a disproven location.

ISP (ASN)	Unknown Location	Disproven Location
Telstra (1221)	0.90%	0.02%
Sprint (1239)	2.17%	0.01%
Ebone (1755)	0.68%	1.19%
Verio (2914)	0.94%	0.20%
Tiscali (3257)	4.30%	0.62%
Level3 (3356)	0.94%	0.98%
Exodus (3967)	0.00%	0.00%
VSNL (4755)	4.17%	0.00%
AboveNet (6461)	2.20%	0.97%
AT&T (7018)	1.47%	0.07%

location represents obvious inconsistency in location assignment. This method may underestimate the number of incorrect locations, particularly for routers that consistently take a long time to respond.

A few details complete the description of this method. First, round trip time estimates were incremented by 1 ms to account for traceroute implementations that truncate the fractional time. Second, traceroutes from seven traceroute servers were discarded. Five traceroute servers had a pattern of improperly formatted output which leads to erroneous data. These errors can be tolerated by filtering out stray and unlikely links, but these malformed traceroutes would "disprove" the location of a router if left uncorrected. The remaining two traceroute servers combined had 131 vantage points that all used clocks that implemented round-trip-time estimation with 4 ms precision.[9] These vantage points would provide a round-trip-time estimate of 0 ms even when the true round trip time could have been 2 ms.

The ISPs having the highest percentage of disproven locations, Ebone and Level3, had 6 and 27 incorrect locations. By looking through a sample of the router names associated with these addresses, the names of at least some of these routers were simply incorrect. For example, a vantage point operated by Qwest in Burbank, California[10] contacted `ge-7-0-0.mpls2.hongkong1.level3.net` in less than 1 ms and contacted `so-1-0-0.mp2.amsterdam1.level3.net` in 9.78 ms. These DNS names are likely incorrect or may have changed during mapping.

This evaluation suggests, but does not prove, that combining DNS names with inferred neighbor locations is a reasonably accurate method for assigning locations. Only a very small fraction of implausible addresses were found, and these appear to be caused by a few incorrect or out-of-date names. One limitation of this evaluation is that it does not quantify the precision with which location is estimated (whether the inferred geographic location is within 10 miles or 100 miles of the actual location). This check will also not find that routers in a "spoke" POP were erroneously assigned to a "hub" POP: because the router in the spoke POP is reached through the hub, it will take more time to be reached, so the analysis cannot "disprove" that the router is in the hub POP unless the spoke POP has a vantage point.

One could extend this verification technique to assign locations to routers based on round-trip-time measurements [91, 136]. This approach would likely require that the measurement platform be as diverse as the traceroute servers used here but have precise round trip time measurements. Here, I used round trip time mea-

[9] These traceroute servers provided gateways to a traceroute implementation on routers.

[10] The analysis guessed Anaheim, California, but the vantage point is chosen by a list box on a traceroute server and named "Burbank," so is likely located there.

surements to find egregious errors: placing routers in incorrect cities. To determine the location of a router among the suburbs of a city by using this method is likely to require vantage points in each suburb that can measure round-trip-times below 1 ms.

5.6 Accuracy of the Resulting Maps

Establishing the correctness of the measured maps is a difficult task because the true maps are not available for comparison. To provide confidence that the measured maps are reasonably accurate, I present the results of two comparisons.

The first comparison was made by a few ISP operators—people familiar with the true design of their networks, but not necessarily able to provide me the complete picture needed for a detailed comparison. Although my focus on individual ISPs was intended to support study of ISP network design, it presented a unique opportunity to compare what I measured to what ISP operators knew. Second, I quantitatively estimate the completeness of the maps by comparing them to the data from Skitter, Route Views, and scans of IP address ranges.

5.6.1 Comparison to ISP-internal maps

Colleagues at three of the ten ISPs compared the Rocketfuel measured maps to their own internal maps. I do not identify the ISPs because their comparison was confidential. Below I list the questions I asked and the answers I received.

1. *Did Rocketfuel miss any POPs?* All three ISPs said *No*. One ISP pointed out a mis-located router; the router's city code was not classified in my undns database (Section 4.3.3).

2. *Did Rocketfuel miss any links between POPs?* All three said *No*, though two identified a spurious link in their ISPs. This could be caused by broken traceroute output or a routing change during the trace.

3. *Using a random sample of POPs, what fraction of access routers were missed?* One could not spot obvious misses; another said all backbone routers were present, but some access routers were missing; and the third said routers from an affiliated AS were included.

4. *What fraction of customer routers were missed?* None of the ISPs were willing to answer this question. Two claimed that they had no way to check this information.

5. *Overall, do you rate the maps: poor, fair, good, very good, or excellent?* I received the responses: "Good," "Very good," and "Very good to excellent."

These results were encouraging, as they suggest that the backbone is nearly accurate and POPs structures are reasonable. Next, I compared my maps to the published ISP backbone maps and found many discrepancies. Restated, the marketing maps published on ISP Web sites are not authoritative sources of topology. Many have missing POPs, optimistic deployment projections, and show parts of partner networks managed by other ISPs.

5.6.2 Comparison to Scans of IP Address Ranges

Next, I estimate the completeness of the measured maps by seeing how well they cover the IP address space used for routers. Most ISPs appear to assign router IP addresses in a few blocks to simplify management and separate traffic for customers from traffic for the network routers. Looking for addresses not discovered in these IP address blocks will help answer the primary question of this section: how much detail was missed?

Scanning for IP addresses is not, in itself, a generally useful mapping technique, because it may find parts of the network that do not actively participate in forwarding. Further, recall from Section 4.1 that probing a large list of addresses owned by an ISP is not a practical method for mapping. It may also miss routers that are not responsive to probing.

The goal of this comparison is to estimate a lower bound for the completeness of the measured maps in discovering all ISP routers. I selected 60, /24 prefixes from each ISP that included at least *two* routers in the measured maps. I chose only prefixes with at least two routers because many prefixes, such as those that connect ISPs at exchange points, have only one router from the mapped ISP. The coverage of such a prefix would be 100%, providing little information. I chose prefixes to ensure that both backbone and access routers were represented.

To quantify the completeness of the mapping of these prefixes, I searched each for interfaces of "new" routers. A scan of a prefix consists of sending a packet to every address to solicit a response. A new router interface is evidence that a part of the topology was not discovered by traceroutes. Because I cannot tell directly if an address that responds to the scan belongs to a router interface, I consider it one only if it has a name that follows the naming convention of the ISP.

I chose criteria for this test so that it would provide a lower bound on completeness. First, a new address may represent a host that does not act like a router by forwarding traffic for others. Second, the percentage comparison applies to addresses and not routers, which will underestimate the completeness of the map. I remove aliases for already known routers, which means this completeness estimate is independent of the accuracy of the alias resolution tool, but the newly-discovered addresses may belong to just a few routers. Removing aliases of known routers is, however, necessary because these aliases may represent one direction not observed of a symmetric link that is already present in the map (see Figure 4.3). That is, these aliases are not likely to represent a loss in completeness in discovering routers.[11]

Table 5.12 shows the estimated percentage coverage for each ISP. Coverage is the number of known IP addresses relative to the total number of addresses seen in the subnets, not counting aliases of known routers. If the ISP has a consistent naming convention for backbone routers and access routers, I break the total into separate columns for each. I use n/a when the ISP does not have a naming convention for that type of router. The results in this table suggest that my mapping techniques find 64–96% of the ISP backbone routers. The access router coverage is fair to good, depending on the ISP.

5.6.3 *Comparison to Route Views BGP Adjacencies*

I also estimate the completeness of the Rocketfuel maps by comparing the peering links they include to the adjacencies seen in Route Views BGP tables [80]. Every adjacency between a mapped AS and a neighboring AS found in the Route Views table should, in a complete router-level map, be matched by at least one link connecting a router in each AS. A limitation is that advertised BGP AS paths do not always match the paths used for IP forwarding [127]. Thus, even a perfect map will not match Route Views completely.

Figure 5.10 compares the number of BGP adjacencies seen by Rocketfuel and Route Views for each ISP I studied. The worst case for Rocketfuel is AT&T (7018), where it still finds a link to more than 63% of the neighboring ASes seen by Route Views. Rocketfuel discovers some neighbors that are not present in Route Views data, a result consistent with that found by Chang, *et al.* [28]. Route Views collects BGP information from different parts of the network than observed with the public traceroute servers; by the nature of BGP (Sections 2.2.1 and 3.2.3), this information is incomplete. An inspection of the discrepancies showed that Route Views contains more adjacencies to small neighbors (those with low degree in the AS-graph), while Rocketfuel finds more adjacencies with large neighbors.

[11] To quantify the completeness of links requires an understanding of the underlying switched vs. point-to-point layer two topology, which is future work.

Table 5.12: Estimate of Rocketfuel's coverage, relative to the responsive IP addresses named like routers, excluding aliases of known routers. The comparison is conservative because not all of these addresses in the comparison set represent routers. "n/a" implies that the ISP's naming convention does not differentiate between backbone and access routers. "Total" includes backbone, access, and customer routers.

ISP (ASN)	Backbone	Access	Total
Telstra (1221)	64.4%	78.1%	48.6%
Sprint (1239)	90.1%	35.0%	61.3%
Ebone (1755)	78.8%	55.1%	65.2%
Verio (2914)	75.1%	60.6%	57.5%
Tiscali (3257)	89.1%	n/a	41.5%
Level3 (3356)	78.6%	77.4%	55.6%
Exodus (3967)	95.4%	59.8%	53.6%
VSNL (4755)	n/a	n/a	48.4%
Abovenet (6461)	83.6%	n/a	76.0%
AT&T (7018)	65.4%	91.6%	78.9%

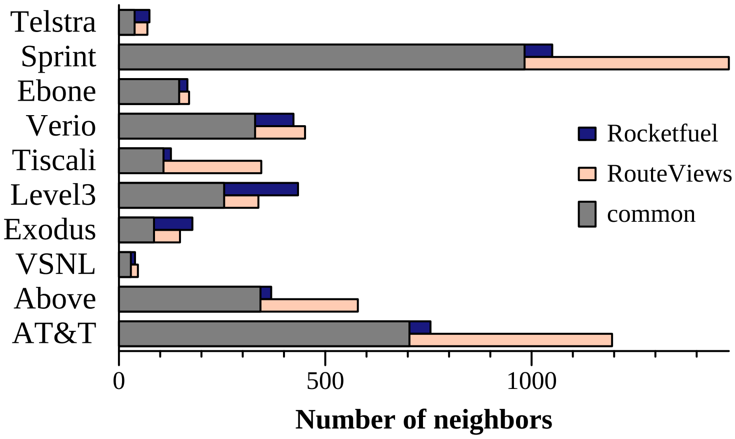


Figure 5.10: Comparison between BGP adjacencies seen in the maps and those seen in the BGP tables from Route Views.

5.6.4 *Comparison to Excerpts of Skitter's Whole-Internet Map*

Skitter is a traceroute-based mapping project run by CAIDA [33]. Skitter has a different goal, to map the entire Internet, and a different approach, that of taking a complete set of traceroutes from a few dedicated servers. Although public traceroute servers cannot immediately be used to map the whole Internet, Rocketfuel maps show that more detail can be found. I analyze Skitter data collected on November 27 and 28, 2001. (Rocketfuel collected data during December 2001 and January 2002.) I compare the two data sets on IP addresses, routers after alias resolution, and links seen in each mapped AS. Figure 5.11 graphs the IP address statistics and Table 5.13 summarizes all three statistics.

Rocketfuel finds six to seven times as many links, IP addresses and routers in the selected ISPs than Skitter does. Because Skitter collects many traces from fewer vantage points, this result implies that more vantage

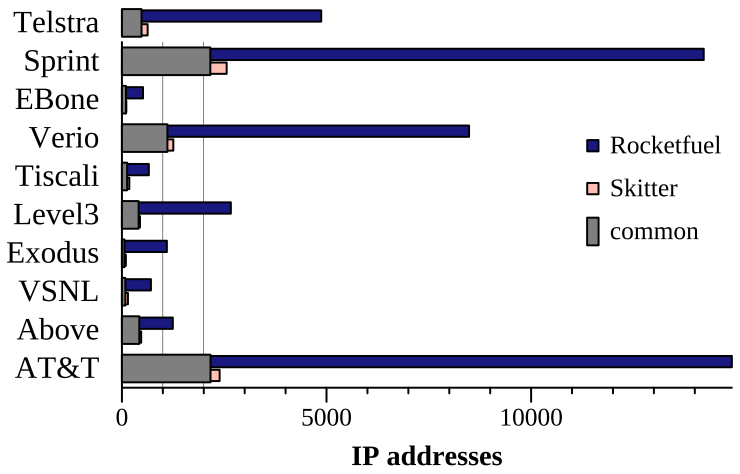


Figure 5.11: Comparison between unique IP addresses discovered by Rocketfuel and Skitter for each ISP studied.

Table 5.13: Comparison of links, IP addresses, and routers discovered by Rocketfuel and Skitter, aggregated over all 10 ISPs. Unique features are those found in only one of the maps. Unique routers are those without aliases in the other data set.

	Links		IP addresses		Routers	
	Total	Unique	Total	Unique	Total	Unique
Rocketfuel	69711	61137	49364	42243	41293	36271
Skitter	10376	1802	8277	1156	5892	870

points contribute more completeness than directed probing and path reductions sacrifice for efficiency. Few routers and links were found by Skitter alone. While some of this difference may be due to the evolution of the Internet during the period between the Skitter and Rocketfuel data collection, some differences are likely the result of Rocketfuel missing some routers. The bulk of the routers found by Skitter but not Rocketfuel either belonged to neighboring ISPs[12] or were access routers.

5.7 Sampling Bias

As described in Section 3.1.3, traceroute can only discover the links along paths from the vantage points of the measurement platform to a chosen set of destinations. When mapping a densely-connected network, this limitation means that links close to vantage points are more likely to be traversed (and thus discovered) than links farther away. This problem has been termed *sampling bias*: the measured topology is incomplete, but measured in such a way that links are not sampled uniformly because those nearer to vantage points are more likely to be observed.

[12] Removing routers from neighboring ISPs requires analysis not performed over Skitter data when names are ambiguous, for reasons described in Section 3.3.

To provide accuracy, Rocketfuel was based on two key ideas: to compose an accurate map would require many vantage points and to map the network in detail would require narrowing focus on a smaller piece. Because it was not clear how many vantage points would be enough (this remains an open question), I chose the largest platform I could find, public traceroute servers that provided 784 vantage points, in the hope that this platform would minimize the effect of sampling bias and yield the most accurate maps possible.

The question addressed in this section is whether the vantage points and measurement method I used were sufficient to remove the appearance of sampling bias. I describe and apply the two tests for sampling bias developed by Lakhina, *et al.* [69], to each of the measured ISP topologies. Despite the positive comparisons in the rest of this chapter, some of the measured maps show variation in degree distribution consistent with sampling bias. However, no previously studied network maps have "passed" Lakhina's tests. Finally, I speculate on the implications of the observations of sampling bias on the usefulness of the maps and on future traceroute-based mapping studies.

5.7.1 Sampling Bias Tests

The key assumption of the tests for sampling bias described by Lakhina *et al.* [69] is that the true distribution of router out-degree is independent of the distance from vantage point to router. I defer for the moment a discussion of whether this assumption is appropriate for ISP mapping. From this key assumption, the two tests for sampling bias are statistical comparisons of the degree distributions of two sub-populations of all vertices (routers) $\mathcal{V}$: the sub-population of vertices near vantage points $\mathcal{N}$ and the sub-population of vertices far from vantage points $\mathcal{F}$. $\mathcal{N}$ includes vertices nearer in hop-count to the vantage point than the median distance; $\mathcal{F}$ includes vertices of median distance or larger. To determine the distance of each router from the nearest vantage point, I re-parse the traceroute measurements: each address within the ISP is mapped to a unique router identifier by alias resolution and each router identifier is mapped to the minimum hop-count (line number in the traceroute output—see Figure 3.1) at which it was seen. Because $\mathcal{F}$ consists of the vertices of at least median distance, that set will be larger than $\mathcal{N}$, that is, $|\mathcal{F}| \geq |\mathcal{N}|$. The likelihood that an arbitrary node is in $\mathcal{N}$ is $p_{\mathcal{N}}$, which will be slightly less than 0.5 because only values less than the median are included.

I apply these tests for sampling bias over the router-level topologies of the ISPs excluding customer and peer routers. Links to customer and peer routers do not count toward the degree of an ISP router. This restricts the test to the pieces of the Internet on which these mapping techniques focused and are expected to be reasonably accurate. Applying these tests to each ISP independently means that each ISP has an $\mathcal{N}$ and $\mathcal{F}$.

Lakhina *et al.* find sampling bias in traceroute studies when the null hypotheses of the following two tests are rejected with high confidence (99.5% or greater). I will continue to use that threshold although this standard may be too high for the relatively small populations of routers in the ISPs I study. The tables of results include sufficient information to modify the analysis for other confidence values.

The first test is whether the 100 routers having the highest degree[13] appear with similar frequency in $\mathcal{N}$ and $\mathcal{F}$. The null hypothesis for this test, $\mathcal{H}_0^{C1}$, is that the high-degree routers are equally likely to appear in $\mathcal{N}$ as they are in $\mathcal{F}$. Of all routers in $\mathcal{V}$, the 100 routers with the highest degree are placed in set ν(or more if there are ties). The expected number of highest-degree vertices in $\mathcal{N}$ is $p_{\mathcal{N}}\nu$. The observed number of highest-degree vertices appearing in $\mathcal{N}$ is κ. Lakhina *et al.* determine how unlikely a large value of κ is by using Chernoff bounds, which determine the likelihood of an outcome that exceeds the mean by a $(1+\delta)$ multiplicative factor using the following equation:

$$\Pr[\kappa > (1+\delta)p_{\mathcal{N}}\nu] < \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{p_{\mathcal{N}}\nu}$$

[13] Lakhina *et al.* use the routers with out-degree in the 99th percentile: there are usually only ten such routers for most ISPs I studied, providing insufficient data for statistical analysis. The choice of 99th percentile routers in the original study was not explained, though may be justified by the study of larger topologies [69]. I adjusted the test for these smaller topologies in consultation with Anukool Lakhina.

Table 5.14: A contingency table for studying the sampling bias in the ISP maps. The second test for sampling bias uses the χ^2 test to see if the distribution of out-degree is independent of whether the router is near or far.

	Routers of out-degree										
Set	<3	<7	<11	<16	<19	<24	<29	<34	<39	≥39	Totals
$\mathcal{N}$	6	28	35	41	28	41	27	26	42	40	314
Expected	24.4	36.4	30.6	33.5	26.3	35.4	31.1	27.3	33.0	35.9	
$\mathcal{F}$	45	48	29	29	27	33	38	31	27	35	342
Expected	26.6	39.6	33.4	36.5	28.7	38.9	33.9	29.7	36.0	39.1	
$\mathcal{V}$	51	76	64	70	55	74	65	57	69	75	656

A probability below 0.005 will be used to reject the null hypothesis that there is no sampling bias (informally, low probability of a high κ indicates sampling bias). If the probability is greater than 0.005, the null hypothesis cannot be rejected: this does not show that sampling bias is not present, only that the analysis is inconclusive.

Lakhina's second test uses the χ^2 statistic to determine whether the sub-population $\mathcal{N}$ is drawn randomly from the larger population $\mathcal{V}$. The null hypothesis for this test, $\mathcal{H}_0^{C2}$, is that the sub-populations are independently drawn from the same distribution. Although Lakhina *et al.* refer to Jain [66, Chapter 27] for the χ^2 technique, they appear to use *contingency tables* as described in common statistics textbooks.[14] I split each of the ISP routers in $\mathcal{N}$ and $\mathcal{F}$ into at least four bins of varying size large enough to have approximately 60 nodes in $\mathcal{V}$. Table 5.14 shows an example contingency table for AT&T routers. Beneath each observed count in $\mathcal{N}$ and $\mathcal{F}$ is the expected number of routers that should appear in the bin if each of these sub-populations is drawn at random from $\mathcal{V}$. The χ^2 value is calculated from the sum of the squares of the differences between the observed and expected number of routers in each bin.

I modified the test slightly for VSNL and Ebone topologies. In both tests, I removed VSNL (4755). VSNL was simply too small to support any conclusions. Ebone showed a peculiar effect: Ebone deployed traceroute server vantage points widely within its network, which made the median distance from the closest vantage point to an Ebone router just 1 hop. Because the near set is defined to include only vertices of distance strictly less than the median [69], the near set was empty. Instead of discarding Ebone, I forced the "median" to be 2 hops, which makes the near set larger than the far set. I believe the analysis depends more on the existence of some value to split $\mathcal{N}$ and $\mathcal{F}$, not that the value specifically be the median.

5.7.2 *Results*

I now present the results of these two tests for sampling bias applied to the ISPs I studied. Results for these ISPs may be of limited statistical significance because the populations are small. I do not show the union of all ISPs; the analysis does not compose well because different ISPs have different underlying degree distributions, range of out-degree values, and different median distances from vantage points.

In Table 5.15, I show the results of the first test, which compares the frequency of νrouters in $\mathcal{N}$ and $\mathcal{F}$. Unlike the network topologies studied by Lakhina *et al.* [69], the null hypothesis of no sampling bias is not always rejected. That is, this test does not show sampling bias in all measurements at the 99.5% confidence level. Interestingly, for AT&T and Ebone, the number of very-high-degree routers in $\mathcal{N}$ is low enough that the null hypothesis would not be rejected with even a 90% confidence level. One possible explanation is that

[14] Textbooks describing the use of contingency tables for finding independent sub-populations include Hogg & Ledolter [57, Chapter 6], Devore [37, Chapter 14], and Walpole & Myers [130, Chapter 10].

Table 5.15: Tests of Lakhina null hypothesis $\mathcal{H}_0^{C1}$. The fraction of nodes in the near set, $\mathcal{N}$ is $p_{\mathcal{N}}$, the number of top-100 routers after breaking ties ν, the number of top-100 routers found to be in the near set is κ, and the Chernoff bound is α. An α smaller than 0.005 would suggest that the null hypothesis (no sampling bias occurs) should be rejected with 99.5% confidence. VSNL (4755) is omitted due to its small size. The median distance for Ebone was 1 because of vantage points inside their network; to avoid $\mathcal{N} = \emptyset$, I set the "median" to be 2, even though it means $|\mathcal{N}| > |\mathcal{F}|$.

ISP (ASN)	median hops	$p_{\mathcal{N}}$	ν	κ	α	Result
Telstra (1221)	6	0.446	111	93	2.56×10^{-7}	Reject (apparent bias)
Sprint (1239)	3	0.416	106	69	0.00255	Reject (apparent bias)
Ebone (1755)	1 (2)	0.547	105	73	0.141	Inconclusive
Verio (2914)	4	0.462	112	88	2.82×10^{-5}	Reject (apparent bias)
Tiscali (3257)	5	0.434	130	87	0.000819	Reject (apparent bias)
Level3 (3356)	2	0.463	106	88	3.88×10^{-6}	Reject (apparent bias)
Exodus (3967)	2	0.423	116	72	0.00943	Inconclusive
AboveNet (6461)	2	0.414	108	75	0.000204	Reject (apparent bias)
AT&T (7018)	4	0.479	109	59	0.652	Inconclusive

the routers owned by each ISP may exist in a range of distances too narrow to observe a dependence between distance and degree.

In Table 5.16, I show the results of the second test, which uses χ^2 to compare the histograms of router out-degree in $\mathcal{N}$ to $\mathcal{V}$. The $\chi^2_{[1-\alpha;l-1]}$ value is taken from Devore [37]. That the calculated χ^2 value is always larger than $\chi^2_{[1-\alpha;l-1]}$ implies that $\mathcal{N}$ and $\mathcal{F}$ are not sampled at random from $\mathcal{V}$. These values are consistent with sampling bias, but the test is quite sensitive and may not have been given enough data for a good result.

To understand the differences between these distributions visually, I present the complementary cumulative distribution functions (CCDFs) of the router degree distributions in both $\mathcal{N}$ and $\mathcal{F}$ in Figure 5.12. This style of graph is commonly used to characterize degree distributions, an analysis which I defer to Section 7.2.1. These CCDFs plot the probability (the y-axis) of finding a router of degree greater than the x-coordinate on a log-log scale. The $\mathcal{V}$ line extends further down because $\mathcal{V}$ has more data points so lower probabilities can be plotted. The $\mathcal{N}$ routers have higher degree (lines further to the right) than those in $\mathcal{F}$. The way to draw conclusions from these graphs is to observe that when the degree distribution of $\mathcal{N}$ and $\mathcal{F}$ are close, there appears to be little sampling bias; when they are far apart, the result may be biased and the $\mathcal{N}$ set may be a more accurate representation of the true degree distribution. Interestingly, the graph of AT&T (7018) distributions shows little bias.

On the whole, these results are consistent with some sampling bias. Because ISP maps are much smaller than the whole-Internet maps studied by Lakhina, these comparisons may not have had sufficient data to be conclusive.

5.7.3 Implications of Apparent Sampling Bias

Because the philosophy underlying these measurement techniques was to use many vantage points to gain completeness in the measurement and avoid sampling bias, it is important to consider the implications of finding sampling biases in some of the resulting maps. In this subsection, I first describe possible causes for the appearance of sampling bias. Then I present a few possibilities for correcting for sampling bias should it appear to be an inherent limitation of using traceroute.

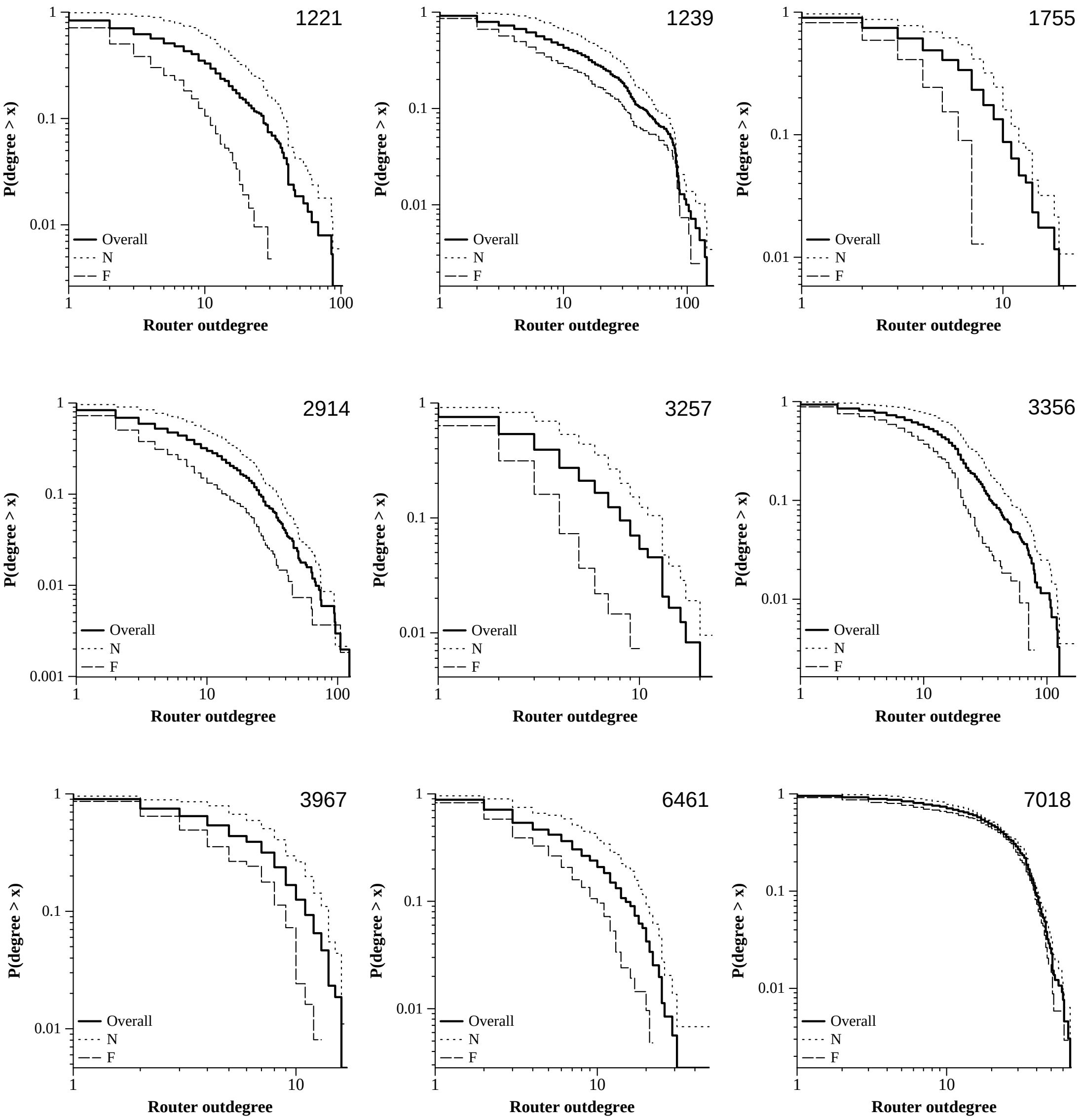


Figure 5.12: Router out-degree CCDF by ISP, excluding customer and peer nodes, broken down into near and far sets. The names of these ISPs can be found in Table 5.1 on page 35. VSNL (4755) is omitted. Similar distributions between near and far suggests limited sampling bias.

Table 5.16: Tests of null hypothesis $\mathcal{H}_0^{C2}$. The calculated χ^2 is larger than the $\chi^2_{[1-\alpha;l-1]}$ for every ISP and for the combined topology, implying sampling bias in the measured topologies. VSNL (4755) is omitted, and the median distance for Ebone was increased to 2.

ISP (ASN)	bins (l)	α	$\chi^2_{[1-\alpha;l-1]}$	χ^2	Result
Telstra (1221)	6	0.995	16.75	147	Reject (bias found)
Sprint (1239)	10	0.995	23.59	160	Reject (bias found)
Ebone (1755)	5	0.995	14.86	48.9	Reject (bias found)
Verio (2914)	12	0.995	26.76	263	Reject (bias found)
Tiscali (3257)	4	0.995	12.84	81.9	Reject (bias found)
Level3 (3356)	10	0.995	23.59	128	Reject (bias found)
Exodus (3967)	5	0.995	14.86	42.5	Reject (bias found)
AboveNet (6461)	5	0.995	14.86	65.0	Reject (bias found)
AT&T (7018)	10	0.995	23.59	43.3	Reject (bias found)

The analysis of this section relies on the assumption that high-degree routers are as likely to be in $\mathcal{N}$ as $\mathcal{F}$. That is, that the true distribution of out-degree is independent of proximity to vantage points. It is unclear whether this should be true of ISPs in the Internet. For example, as I show in Section 7.2.2.1, a few POPs are very large; large POPs are likely to be densely connected (so routers have high degree) and likely to have more vantage points nearby (because more addresses are reached). Smaller POPs with fewer routers will have routers of smaller degree and be less likely (perhaps much less likely) to have nearby vantage points. It is unclear whether this imbalance would cause the appearance of sampling bias where none existed.

I believe that the main cause of apparent sampling bias is the use of switched networks in POPs that lack vantage points. When a switched network is present, the graph should be a fully-connected “clique” of routers attached to the link, but when no vantage point is present, it will at best appear to be a star in which most routers are not directly connected to each other. This suggests two ways to avoid bias in the future: have more vantage points or explicitly detect and correct for switched (multiple-access) links. Adding more vantage points is straightforward but requires deployment. I describe future work in detecting switched links in Section 8.2.2.4. If a switched link is detected, the “clique” of connectivity in the router-level topology can be inferred, even though the connections between access routers may not have been directly measured by traceroute.

The term “sampling bias” hides the underlying causes for missing distant links: links are not missed by random chance that varies with hop count, links are missed because no observed path traverses them. One might develop a network mapping approach that, in addition to finding links that do exist, infers that two routers are not directly connected because a multi-hop path has been observed to connect them. Such an analysis may occasionally be incorrect, for example if a low-capacity link does connect two routers but is unused. However, comparing three sets of router pairs—those with discovered links, those with inferred no-links, and those with unknown connectivity—may lend insight into where sampling bias occurs and how to avoid it in analysis of the maps. I do not perform this analysis here: enumerating the links that are not present is a computational challenge.

5.7.4 Summary

In this section, I found evidence of sampling bias in some of the measured ISP maps. Some of the maps I measured, such as those of AT&T and Ebone, show sampling bias in only one of the two tests and show relatively little sampling bias in the CCDF graphs. No prior measured maps have been found to be without sampling bias. The maps I measured represent a step toward measuring bias-free topologies, but further work remains to eliminate sampling bias.

5.8 Summary

I evaluated the efficiency and accuracy of the individual methods used to collect the maps as well as the accuracy of the resulting maps. The key is to be aggressive at eliminating redundant measurements in both traceroute collection and alias resolution because all-to-all probing in both is ultimately impractical.

A combination of techniques can eliminate redundant measurements. BGP directed probing and path reductions combine to reduce the number of measurements required to one-thousandth of what would be chosen by an all-to-all mapping approach while eliminating few useful probes. Techniques to guide the search for IP address aliases can disprove over 98% of the possible alias pairs without further measurement.

Even though few traceroutes are chosen, the resulting maps are reasonably accurate. Comparisons made by some ISP operators to the real maps show that they are reasonably accurate; quantitative comparisons show that they are at least within a factor of two of the complete topology and are already many times more complete than prior efforts. However, some sampling bias remains in the maps and work remains to improve the completeness of measured ISP topologies.

Chapter 6

NETWORK ROUTING POLICY INFERENCE

In this chapter, I describe how to infer network routing policies from the combination of measured network topologies with the traceroute paths used in their measurement. I evaluate the inferred intra-domain and peering routing policies by how well they predict observed paths through the network.

Network routing policy, like network topology, is an expression of design goals. Within a network, routing policy can be used to improve performance without new hardware. Across networks, routing policy can implement cooperative designs that improve global performance or share the cost of operating networks. In light of the diversity of network topologies, one might expect similar diversity in network routing. Because routing policy is expressed through the configuration of software, understanding the diversity of routing policies may shape the design of simpler routing protocols or the creation of best-practices.

Of the three levels of routing policy, I focus on the lower two: intra-domain and peering. The third, inter-domain routing, has been studied by others [49]. Measured router- and POP-level maps provide a means to explore the lower levels of network routing policy.

The key insight underlying this work is that paths *not* taken expose the choices of routing policy. A complete topology includes many alternate paths; that a path was used instead of an alternate exposes a routing policy decision. Aggregating observations of these decisions can expose a pattern of routing policy. Recognizing these patterns is the subject of this chapter.

I use the following question as an evaluation metric for both inferred routing policies: are they predictive? Although the primary goal of routing policy inference is to characterize and understand, the ability to predict paths is evidence of the accuracy of the inferred policies. Some observations will not match predictions, however, because the inferred routing policies are of steady-state behavior. Routes during failures and changes may not be predicted correctly. Incorporating observations of link failures and topology changes into the inferred network routing policy is future work.

To evaluate the methods for inferring routing policy, I use two data sets. To gain confidence in the inference of intra-domain routing, I reuse the ISP maps from Chapter 5. I evaluate intra-domain routing policy inference using these measured, router-level topologies to show the scalability of the techniques. To build upon the inferred intra-domain routing policy and evaluate how well the techniques of this chapter infer peering routing policy, I collected a new data set spanning 65 ISPs. This larger data set supports the analysis of the relationships between many more pairs of ISPs. In particular, it supports analyzing the relationships between tier-1 and tier-2 ISPs.

In this chapter, I first describe how to infer intra-domain routing policy. I then evaluate the accuracy of inferred routing policies by how predictive they are: whether a routing policy can be derived from a subset of the measurements and then predict the remaining paths. I then describe the dataset collected for inferring peering routing policy. Finally, I describe and evaluate the inference of peering routing policy.

An analysis of the prevalence of different routing policies is deferred to Chapter 7.

6.1 Intra-domain Routing Policy Inference

Intra-domain routing policy is usually expressed through the assignment of weights to links. I assume that the ISP uses shortest weighted path routing because shortest weighted paths are chosen by all common intra-domain routing protocols, including OSPF [82], IS-IS [90], EIGRP [31], and RIP [56]. To infer intra-domain routing policy, I derive link weights consistent with (that is, result in the same set of observed paths as) the actual weights configured by the ISP.

In this section, I describe my method for inferring link weights. I first present the basic method based on linear programming. I then describe how to identify redundant constraints to make it practical. I apply a technique called constraint hierarchies to tolerate some error in measurement. I describe how to increase the completeness of the observations of path data to provide more observed paths to the constraint system. Finally, I discuss the limitations of the link weights derived using this method and the applicability of the assumption that the ISP uses shortest-weighted path routing.

6.1.1 Initial Method

Consider a network with vertices (nodes) and directed edges (links). Each edge has a positive weight configured by the network operator, and the weight of a path is the sum of the weights of its edges. The least weight path, or paths in case of a tie, between two vertices is *chosen*. A *chosen* path is one that has been selected by the operator. An *observed* path is what appears in a traceroute in the data set. All chosen paths will be observed if the measurement is complete, and only chosen paths will be observed if failures and transients do not occur. Because neither of these can be relied upon, I address transient paths and measurement errors in Section 6.1.3 below.

The goal of intra-domain routing policy inference is to assign weights to the edges such that the assignment is consistent with observations—only the least weight paths between all vertex-pairs are observed. This abstract problem is an instance of the inverse shortest paths problem (ISPP) [26, 44]. There are many valid weight assignments for two reasons. First, scaling all weights by the same factor does not change which paths are shortest. Second, changing the weight of a link by a small amount may have no observed effect on routing. The method I describe will produce one feasible set of edge weights, not the one in use by the ISP.

The key insight underlying routing policy inference is that paths not taken expose routing policy choices. That is, the many alternate paths present in a reasonably complete network topology provide a rich source of data to explain why a specific path was observed. To apply this insight to infer link weights, I make two observations: $i)$ the weight of the chosen path is less than that of other possible paths between the same vertices, and $ii)$ multiple chosen paths between two vertices have equal weight.[1] My method for inferring link weights is to translate these observations for every observed path into constraints in a constraint system. I will use an example to illustrate the constraints generated for one observed path.

Consider the network in Figure 6.1. To simplify this example, all links are directional; in the actual analysis, links are bidirectional. Assume that the only observed paths between A and G are ADG and $ABEG$. Applying the observations above, these paths have equal weight and they are shorter than all simple alternate paths ACG, $ACFG$, $ABDG$, $ADEG$ and $ABDEG$. These facts can be represented by the following constraints:

1. $w_{ad} + w_{dg} = w_{ab} + w_{be} + w_{eg}$
2. $w_{ad} + w_{dg} < w_{ac} + w_{cg}$
3. $w_{ad} + w_{dg} < w_{ac} + w_{cf} + w_{fg}$
4. $w_{ad} + w_{dg} < w_{ab} + w_{bd} + w_{dg}$
5. $w_{ad} + w_{dg} < w_{ad} + w_{de} + w_{eg}$
6. $w_{ad} + w_{dg} < w_{ab} + w_{bd} + w_{de} + w_{eg}$

I repeat this process for the observed paths between every vertex pair to construct the system of constraints. I do not create constraints using circular alternate paths: each circular alternate path has a shorter, simple path

[1] The use of multiple paths simultaneously when paths have equal cost, instead of breaking ties arbitrarily, is called *equal cost multi-path routing*.

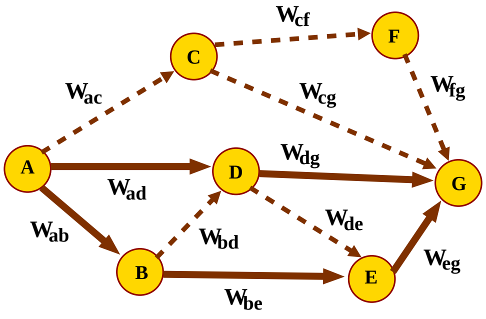


Figure 6.1: An example weighted intra-domain topology. The chosen paths, ADG and $ABEG$, are shown with solid lines, and the alternate paths with dashed lines.

if the cycle is removed, and this simple path is already present in the constraint system. Because the number of vertex pairs and simple paths is finite, the number of constraints is finite.

I solve the constraint system using integer linear programming [83] to obtain link weights consistent with the observed routing. As long as the observed routing is shortest path with (unknown) positive link weights, I can obtain a consistent (solvable) constraint system. The constraint solver [16] implicitly includes other facts in addition to those explicitly relating observed and alternate paths above: that all weights are positive integers and that weights are minimized. These two further constraints mean that when given a set of inputs the solver reaches a unique solution.

Two problems remain. To be practical, the constraint system must be simple enough to be solved by current hardware: specifically, the number of constraints should be polynomial, not exponential, in the number of vertices. This approach must also deal with transient network paths and other errors in the observations. I address these two problems in turn.

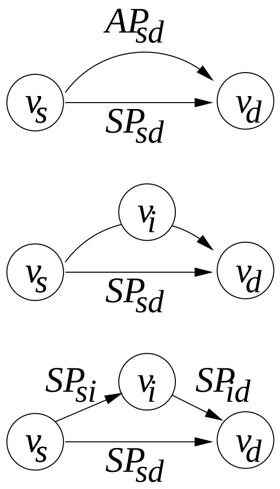
6.1.2 Eliminating Redundant Constraints

Unfortunately, the initial solution leads to an exponential number of constraints because a constraint is created for every simple (acyclic) alternate path. This makes it intractable for large networks. The constraints generated for many of these alternate paths are redundant because only a few alternate paths bound the cost of the chosen path. By removing redundant constraints, the solver can calculate weights for some (though not all) of the larger networks in the Rocketfuel dataset. I now describe how to identify redundant constraints, using Figure 6.2 as an illustration.

Let SP_{sd} be an observed path of vertices from v_s to v_d, and AP_{sd} be an alternate path (never observed) between the same two vertices. Let $W(P)$ denote the weight of path P. The initial solution described above would create a constraint $W(SP_{sd}) < W(AP_{sd})$ for every simple alternate path to express that observed paths are shorter than alternate paths.

The alternate path AP_{sd} is redundant if a shorter path can be constructed by substituting an observed path. More precisely, the constraint $W(SP_{sd}) < W(AP_{sd})$ is redundant if at least one of the following two conditions is true for *any* intermediate vertex v_i in the alternate path AP_{sd}: *i)* AP_{si}, the prefix of AP_{sd} to vertex v_i, is not an observed path between v_s and v_i though another path has been observed between v_s and v_i; or *ii)* AP_{id}, the suffix of AP_{sd} is not an observed path between v_i and v_d though another path has been observed between v_i and v_d.

If SP_{si} is not a prefix, or SP_{id} not a suffix, of AP_{sd}, there is a shorter AP'_{sd} that uses an observed path (SP_{si} or SP_{id}) for the part of AP_{sd} that was not observed. Because observed paths are shorter than alternate



Observed and alternate paths between v_s and v_d. An alternate path (AP_{sd}) is at least as long as an observed path (SP_{sd}).

v_i is an intermediate node along the alternate path.

An alternate path is redundant if, for any intermediate vertex v_i, there is an observed path SP_{si} or SP_{id} that is not a prefix or suffix of AP_{sd}.

Figure 6.2: Eliminating redundant constraints. An alternate path is redundant if the source or destination is connected to an intermediate vertex by a different, observed path. If the alternate path has an observed "shortcut," a shorter path can be constructed making the alternate path redundant.

paths, $W(AP'_{sd}) < W(AP_{sd})$. As a result, the constraint $W(SP_{sd}) < W(AP_{sd})$ can be removed because it is redundant with the constraint $W(SP_{sd}) < W(AP'_{sd})$ which remains in the system.

Based on this rule, Constraints 3 through 6 in the example above are redundant if the observed path between each pair of directly connected vertices is the edge connecting them and the shortest path between B and G is BEG.

A few details complete the description of this approach. First, there may be an intermediate vertex along AP'_{sd} that makes this constraint redundant as well. Second, information is not lost in this process because other constraints will relate SP_{si} to the prefix of AP_{sd} and SP_{id} to the suffix: SP_{si} and SP_{id} are observed paths with their own constraints. Finally, this approach relies upon the existence of an SP_{si} or SP_{id}: if these observed paths are not present, the alternate path cannot be shown to be redundant at that intermediate vertex.

The number of remaining constraints is polynomial in n, the number of vertices.[2] The maximum number of alternate paths for a vertex pair (v_s, v_d) is n, because each vertex v_i can appear in at most one alternate path—the concatenation of observed paths SP_{si} and SP_{id}. If there is more than one chosen path between v_s (or v_d) and v_i, only one constraint is needed because all have equal weight (another constraint will ensure that the two observed paths have equal cost). With n^2 vertex pairs, each with at most n alternate paths, the number of constraints is $O(n^3)$.

The actual number of constraints is much smaller than n^3 because far fewer than n alternate paths are valid. There may be no path that traverses a particular vertex: for example, informally, there should be no useful alternate path between two routers in Seattle that traverses a router in Atlanta.

To find redundant constraints requires many observed paths (SP_{sd}, SP_{si}, SP_{id}). Above, the observed paths SP_{si} or SP_{id} must be known to flag an alternate path as redundant. If no path from v_s to v_i has been observed, the constraint reduction technique must preserve possibly redundant constraints. Because observed paths contribute constraints *and* help to show others to be redundant, it will be difficult to evaluate the predictiveness of the model in the next section: removing observed shortest paths affects not just the correctness of inferred weights but the speed with which they can be inferred.

[2] This limit is false in the pathological case when *every* path is observed between vertices and the equal cost constraint must be created for each. The pathological case is unlikely (if not impossible) with shortest path routing.

Not all paths must be enumerated before pruning out redundant ones. Instead, a dynamic path growing algorithm produces only non-redundant alternate paths. The path growing algorithm iterates the following procedure. Initialize the set of non-redundant alternate paths $\mathcal{P}_1$ with all single-vertex paths: all routers. To construct all non-redundant paths at the next length, $\mathcal{P}_{i+1}$, select each path p in $\mathcal{P}_i$ and extend p with each neighbor of the last router to construct a set of candidate paths $\mathcal{C}$. Add into $\mathcal{P}_{i+1}$ only those paths from $\mathcal{C}$ that both have no cycle and are not redundant using the criteria above. The construction of candidate paths is applied for every path p in $\mathcal{P}_i$, and paths are extended until $\mathcal{P}_j = \emptyset$. The non-redundant alternate paths are thus the union of all $\mathcal{P}_i$. This algorithm constructs all non-looping paths, and prunes out redundant paths during the path growing process.

6.1.3 *Constraint Hierarchies*

Correct inference of routing policy from observed paths is difficult because transient events, such as failures, may cause paths longer than the stable shortest paths to be observed. Constraint hierarchies provide a means to discount measurements of transient events.

Paths observed during transient events cannot be allowed to influence the inference of routing policy: an inconsistent constraint system may result. Rather than recognize and filter such paths before they are passed to the analysis, a constraint system can be constructed to discount transient events. Constraint hierarchies allow a constraint system in which not all constraints can be satisfied simultaneously [21]. The main idea is to associate an error variable with each constraint, and instruct the solver to minimize the weighted sum of errors. Applied to routing policy inference, an error variable is associated with each observed path, and this error variable is weighted by the number of times a path was observed. Because I have now introduced weighted errors, to avoid confusion with weighted paths, I will be careful to specify the distinction by using “error-value weight” or “link weight” to avoid ambiguity.

I now describe the use of constraint hierarchies by an example. In Figure 6.1, again assume that two paths were observed from A to G: ADG and $ABEG$. Because observations may include paths that were observed during transient events but not chosen, these observed paths may not be shortest. I associate error variables with each path, e_{adg} and e_{abeg}, which represent the potential excess weight (beyond that of the true shortest path) of these paths. That is, how much more weight should be given to the path to compensate for the (erroneous) observation that the observed path is shorter than the (unseen) shortest path. Each error value must be non-negative, by design of the constraint solver [16]. The modified constraints are:

1. $w_{ad} + w_{dg} - e_{adg} = w_{ab} + w_{be} + w_{eg} - e_{abeg}$
2. $w_{ad} + w_{dg} - e_{adg} < w_{ac} + w_{cg}$

Similar modifications to the constraints are made for all vertex-pairs. I find a solution consistent with the observed paths yet tolerant of observations of transient paths by minimizing the weighted sum of the error variables. Because the number of times the path was observed is the error-value weight, the solution will be consistent with the most observations. The paths that were not shortest will have positive error variables in the solution.

6.1.4 *Incomplete Information*

A second challenge in using measurement data is that the data may not include observations of all chosen paths. Chosen paths may not be observed in two cases: first, no chosen path between a pair of vertices was observed; second, an alternate chosen path of equal cost was not observed. *Pair-wise completeness* is the fraction of vertex-pairs with at least one observed path. Poor pair-wise completeness blunts the constraint reduction rule described above: it cannot eliminate some alternate paths when observed path information is not available between vertices of the alternate path. This means that when pairwise completeness is low, the number of constraints is no longer polynomial in the number of vertices.

Pair-wise completeness can be improved using two techniques. First, if paths have *reverse path symmetry*, that is, both directions of a link have equal weight, the reverse path of a shortest path is also shortest. I confirmed reverse symmetry for all the ISPs I studied: the most common path from A to B is the reverse of the most common path from B to A.[3] Second, the *optimal substructure* property—any fragment of a shortest path is shortest—gives shortest paths between every pair of vertices in a shortest path.

The second type of incomplete measurement information is that some chosen paths having equal cost may not be observed. The cost of an alternate path is thus not necessarily strictly greater than the cost of the observed path: because the alternate path may be simply be unobserved, the *less than* in the example constraints above are expressed instead as *not greater than*. Restated, observing a path implies that no other path is shorter, not that all other paths are longer.

6.1.5 Limitations

This approach is applicable for any network in which the chosen path between two vertices is determined solely by a single setting of link weights. Most ISP networks use shortest weighted routing, but some features intended to help the protocols scale may allow paths to be chosen that are not least cost. On such protocol feature is BGP confederations [129], in which a globally-visible autonomous system can be internally split into several private ASes—using BGP to route within an ISP. By using BGP, the costs of paths inside these private ASes are hidden, so the shortest path may not always be chosen. Another such protocol feature is OSPF areas, which helps the link-state routing protocol scale by creating a hierarchy of smaller networks called “areas.” OSPF areas allow different routers to have different views of the network topology, contributing two ways shortest weight paths are not always chosen [82]. First, the costs to reach two adjacent subnets must appear to be the same externally when using OSPF area aggregates [105], in which adjacent subnets in the same area are aggregated (merged to form a larger subnet before being advertised, as in Section 2.1.3). Second, if a shortest path is external to an area, it will not be chosen because paths between vertices in the same area are restricted to that area. Such shortest paths are unlikely because weights within an area (likely a POP) are expected to be smaller than weights across areas. When enabled, these features may choose some paths that are not shortest weight and may have some effect on the routing and thus on routing policy inference.

If shortest weighted routing is not used, perhaps through the use of MPLS, link weights inferred by the procedure I will describe may or may not match observed paths. If they do, the underlying mechanism used to achieve shortest path routing is unimportant. However, if the shortest weighted routing is not used and the ISP chooses a set of paths that cannot be expressed through link weights, the inferred link weights will not match observed paths. If inferred weights do not match observed paths, that suggests that the link weight model is inappropriate.

If a network uses circuit technologies the routing policy may not be visible at the IP layer. Inferred link weights are of little value on an MPLS network that does not decrement the TTL;[4] assigning the same weight to every MPLS tunnel would likely be consistent with observed, IP-level routing. As I evaluate the inferred link weights, I will not use Level3’s network for this reason: any model of IP routing will perform well when every pair of nodes is directly connected.

Link weights inferred by this process are not the true weights assigned by ISP operators. The importance of this limitation is easily underestimated, in part because the true weights would provide a rich source of data

[3] In contrast, inter-domain routes are frequently asymmetric.

[4] Recall from Section 2.2.3 that MPLS may be used in one of two modes: decrementing the TTL, in which traceroute can observe the routing but the routing is not shortest path, and not decrementing the TTL, in which the topology appears to be fully connected and the underlying routes are not visible.

into the underlying provisioning and engineering of the network [46]. It remains to be shown whether inferred weights have the same power to predict link capacity, congestion, and latency.

Although I will show in the following section that inferred weights predict routing even of paths not directly observed, their relationship to the operator-assigned weights is undefined. As a trivial example, the solver may infer any value for the weight of a link for which no alternate path exists. As a more complex example, one might expect that the true intra-domain link weights of backbone links between POPs are larger than the weights of links within a POP. Although this may be a common operational practice, its use cannot be detected using inferred weights. The constraint solver need only ensure that the cost of paths within a POP is less than the cost of an alternate path out a backbone link and back: the inferred weight of a backbone link could be as little as one half the weight of a link within the POP.

Finally, inferred link weights of false links will be unusually high. That the highest inferred weights may not be for real links makes questionable an analysis of the range of link weights, even though such an analysis would indicate the expressiveness required for intra-domain routing.

In sum, inferred link weights can be used to predict intra-domain routing (Section 6.2), to understand early- and late-exit peering routing policies (Section 6.4), and to characterize all paths through the ISP network (Section 7.3.1). The precision of inferred link weights does not support their use in inferring link capacities or recovering higher-level ISP link-setting policies.

6.2 Intra-domain Routing Model Accuracy

To evaluate the recovered link weights, I compare the prediction accuracy of the shortest-weighted path model to that of simpler models based on geographic distance and hop count. To be useful, the recovered link weights should be more predictive of unobserved paths than simpler models that are more easily computed. The simpler models I consider are:

- *Latency*, in which the geographic distance between endpoints is used as an estimate of link latency, and that latency is used as the cost of the link. This models a network designed to minimize the time each packet spends traversing links in the network.

- *Hops*, in which each link has a cost of 1. This models a network that minimizes the number of links traversed by a packet. Many paths will have equal length in this network model.

- *Hoplat* is a balance between the two approaches. Because highly meshed topologies have many paths of equal length in hops, this policy uses latency to break ties.

I now evaluate how well each metric—latency, hops, hoplat, and inferred weights—predicts observed routing. Recall that, due to transient failures and measurement errors, perfect prediction accuracy is not expected.

This evaluation is applied to six of the ten router-level datasets from Rocketfuel. Four ISPs were removed from this analysis. Level3 has a logical mesh topology based on some form of layer-2 switching system like MPLS: any model of intra-domain routing would work well because paths are selected at a lower layer. VSNL was too small to study because it appeared to have only a few POPs. AT&T and Verio were too large for the analysis as run on a dual Pentium III, 1 GHz with 4 gigabytes of memory.

Three metrics evaluate the prediction accuracy of the routing model.

1. **All Observed Paths**

   The observed paths metric counts each traceroute path as a data point. With an accurate routing model, most observed paths should be least cost. This measures overall data fit; each path is weighted by the

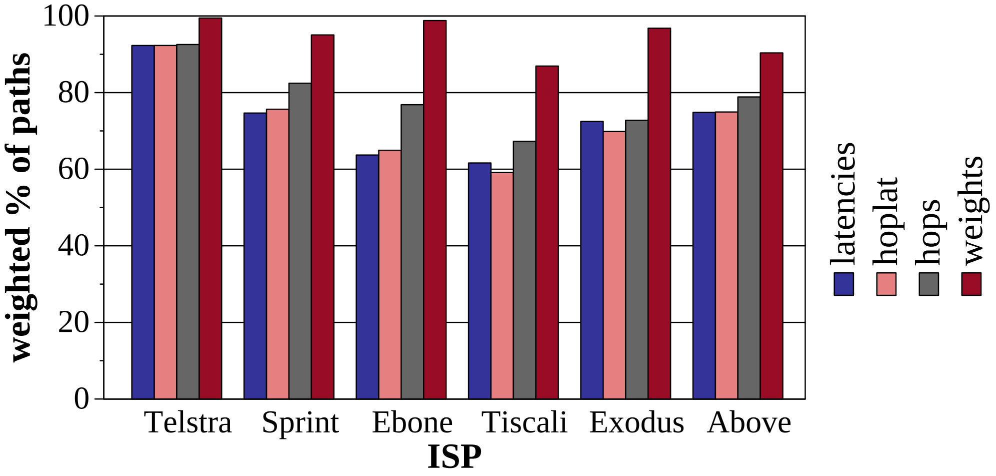


Figure 6.3: Percentage of *observed* paths that were least cost. For each of the policy models on each of the ISP topologies, this graph shows the percentage of traceroute paths that match the predictions of the routing model.

number of times it was observed. Because constraint hierarchies model paths observed more often more accurately, this evaluation should be favorable.

Figure 6.3 shows the fraction of observed paths that were least cost using each metric. Observed paths agree best with inferred weights: while weights fit 87–100% of the paths across the six ISPs, the next best metric (hops) fits only 67–92%.

2. **Dominant Paths**

   A *dominant path* is the most commonly observed path between two nodes, and thus is most likely to be the stable path between them. The dominant path metric counts each vertex pair as a data point. Paths seen less often than the dominant path, such as those observed during failures, are thus weeded out. However, dominant paths between rarely-seen vertex pairs may not be modeled accurately because the constraint hierarchies approach prefers accuracy in most observations. With a good model, most dominant paths should be least cost. All dominant paths are weighted equally, so unlike the evaluation by observed paths, the results are not biased by a few commonly observed paths.

   Figure 6.4 shows the fraction of dominant paths that were least cost. As before, inferred weights match more paths than the other models. While the inferred weights fit 76–98% of the dominant paths, the best alternate metric (hops) fits only 49–82%.

   That the models appear to predict dominant paths less well than observed paths is a result of the aggregation. A few, correct dominant paths observed many times increases the fraction of observed paths that are correctly modeled in Figure 6.3. An incorrect dominant path is likely not to have been observed often, appearing insignificant in Figure 6.3 but prominent in Figure 6.4. Nine out of ten dominant paths that were not fitted by weights were observed five or fewer times.

3. **Routing Between Pairs of Nodes**

   If the routing model predicts multiple least-cost paths between two nodes, all should be observed due to load balancing across equal-cost paths. Neither of the previous two metrics penalizes a model that

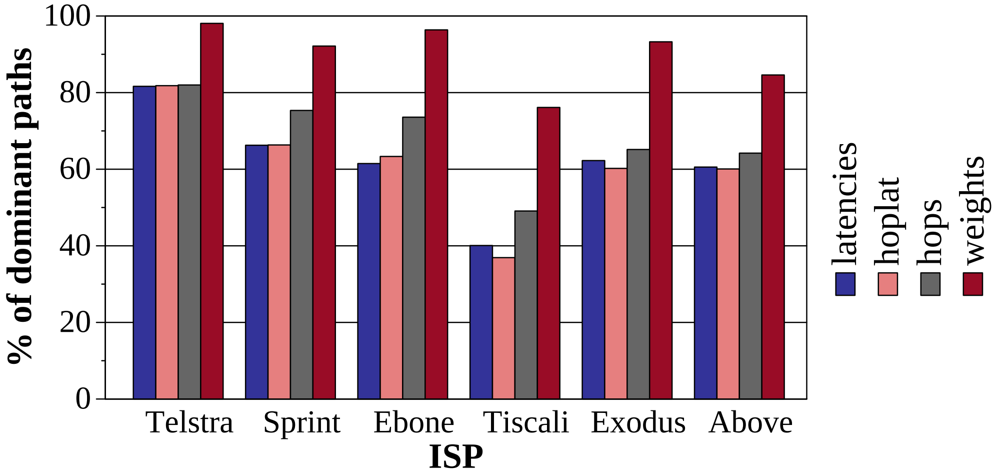


Figure 6.4: Percentage of *dominant* paths that were least cost. For each of the policy models on each of the ISP topologies, this graph shows the percentage of dominant paths that match the predictions of the routing model.

chooses too many shortest paths. A complete characterization of routing means that not only must most observed paths between node-pairs be least cost, but also that paths not observed be more costly. A coarse model that assigns the same cost to many paths would fare well in the first two evaluations by choosing too many least cost paths, but not in this evaluation unless those paths were observed.

To capture how well routing between node-pairs is characterized, I partition node-pairs into the following classes:

- *full*: every least cost path between the pair was observed;
- *partial*: some, but not all, least cost paths were observed;
- *none*: no least cost path was observed.

Figure 6.5 shows the fraction of node pairs in the full and partial classes. Hops is a coarse metric; it partially characterizes 4–20% of the node-pairs, which means that it overestimates the extent of multi-path routing by predicting more least-cost paths than are in the network. This overestimation of multi-path routing accounts for the success of hops in earlier evaluations. Latency can partially characterize routing when paths share a sequence of POPs but differ at a router-level. The alternate metrics fully describe routing only for 47–81% of the node-pairs. Inferred weights fully describe many more, 84–99%, of the node-pairs, and partially characterize only 1–3%. Some partial fitting would also result from not observing enough paths to see all the least cost paths.

6.2.1 Predictive Power of Inferred Weights

I next investigate the predictive power of the model: how many observations does the analysis need to derive link weights that predict the paths not directly observed. Because I do not measure all ISP routes, the closest experiment I can perform is to assess how well weights inferred from a fraction of measurements predict the entire set of route observations. To take a fraction of the measurements, I randomly varied the number of vantage points used to collect the traceroute data, in subsets of size 1%, 5%, and 10–100% in steps of 10%.

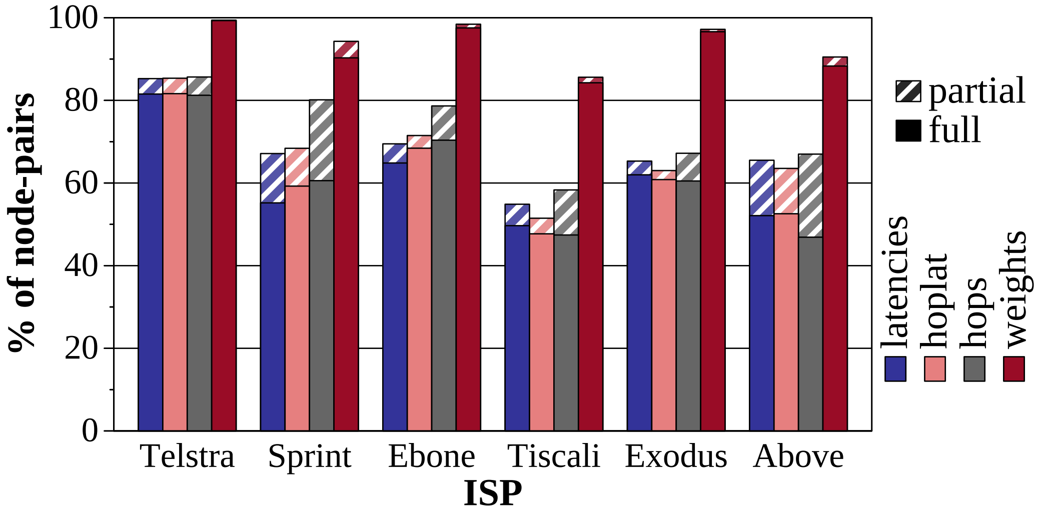


Figure 6.5: Percentage of node-pairs in the full or partial classes. For each of the policy models on each of the ISP topologies, this graph shows the percentage of node-pairs for which every least-cost path was observed (full) or some least-cost paths were observed (partial).

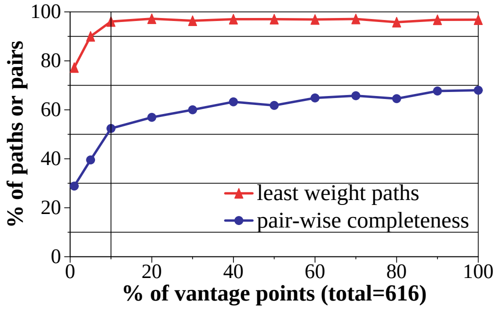


Figure 6.6: Predictive power of routing model for Exodus.

Figure 6.6 shows for Exodus (3967) the percentage of paths in the total dataset that were least cost using the weights inferred from the subset of measurements. The weights derived from 10% of the vantage points describe routing almost as well as weights derived using all of them. For comparison, the figure also shows pair-wise completeness (defined in Section 6.1.4) as a function of the percentage of vantage points. The fit does not improve as pair-wise completeness increases from 50% to almost 70%. This result is encouraging because it suggests that 50% completeness is sufficient to predict paths not directly observed.

Similar pairwise completeness was needed for a fit with good predictive power for other ISPs as well: 80% for Telstra (1221), 40% for Ebone (1755), 50% for Tiscali (3257), and 45% for Abovenet (6461). The high completeness percentage required for Telstra is a consequence of its simple topology, in which the pair-wise completeness was 75% even with just 1% of the vantage points. I could not run this experiment for Sprint (1239): as I reduced pair-wise completeness, few constraints could be eliminated (see Section 6.1.2), and the resulting constraint system was too large.

6.3 Measurements for Studying Peering Routing Policy

In this section, I describe how to construct a large, POP-level map to study peering routing policy. My strategy is to trade the precision of the router-level ISP maps measured earlier for breadth across all of the major ISP networks and a diversity of smaller networks. As described in Section 2.1.2, a POP-level topology bridges the gap between the high-level inter-domain topology of ISPs and the low-level topology of routers. It is the layer at which peering routing policy is most easily observed and studied because the router-level detail is not particularly important: traffic engineering decisions are made at the POP-level [17, 124].

I first describe how I chose 65 ISPs for study. Then, I describe the traceroute dataset I collected using Scriptroute [116] on PlanetLab [97]. I apply the undns library described in Section 4.3 to transform the map of IP-level connectivity from traceroutes into a POP level map. Finally, I show that the intra-domain policy inference method described above applies well to the POP-level intra-domain topologies measured in this section, so that the policy-annotated POP-level topologies can be used as a foundation for understanding peering routing policy.

6.3.1 Choosing ISPs to Study

I used three criteria for choosing ISPs: $i)$ each ISP should be large enough to have interesting intra-domain and inter-domain choices; $ii)$ each ISP should carry enough diverse traffic (be a transit provider) so that its topology and routing policy are easily observable using traceroutes (edge ISPs are much harder to observe); and $iii)$ the set of ISPs should be diverse in size and geographic presence to show any resulting differences in topologies and policies.

Studying all the ISPs in the Internet is desirable, but infeasible for two reasons. First, determining the POP-level structure of each ISP requires a separate template for the naming convention of each ISP; because templates are constructed manually, it is difficult to scale to very many ISPs. Second, some of the analyses I present would have become intractable over very large graphs. Third, it is difficult to use traceroute to cover very small ISPs.

I used the following informal method for selecting 65 ISPs for detailed measurement. I first classified each ISP in the BGP tables to its *tier* using Subramanian's method [121]. Tiers represent how close an ISP is to the Internet "core": tier-1 ISPs are the closest (see Figure 2.1 on page 8). I selected all 22 tier-1 ISPs and 32 high-degree tier-2 ISPs. The degree of an ISP is how many neighboring ISPs it has in the inter-domain topology; I use Route Views BGP tables to estimate the degree of each ISP. Because the degree of an ISP is correlated to its size [29] and geographic extent [70], high-degree ISPs are more likely to have interesting routing policies. Finally, I added 11 tier-3 ISPs. The ISPs I study are listed in Tables 6.1, 6.2, and 6.3. These tables show that the selection process satisfied two of the criteria listed above: the chosen ISPs have diverse degrees and are geographically diverse. (That the ISPs are large enough to have interesting routing choices cannot be determined from the tables.) After mapping these ISPs, I discovered that a few have a very small or no backbone, providing another dimension in ISP diversity.

Even though I study a tiny fraction of ISPs in the Internet, this chosen subset is useful for studying network routing policy because it includes most large providers in the network. Combined, these ISPs account for 40% of the singly homed, globally routed IP address space, implying that a significant fraction of Internet traffic traverses this set of ISPs. In fact, 97.7% of traceroutes in the dataset, collected “blindly” in that they are not guided by BGP, traversed at least one of these 65 ISPs. I next describe how these traceroutes were collected.

6.3.2 Traceroute Paths

The goal of this data set is to support the study of peering routing policy. Peering routing policy is affected by BGP configuration parameters of MEDs and localpref, described in Section 2.2.2, which apply to BGP-advertised prefixes. Rocketfuel, which focused on limiting measurements while using public traceroute servers to map a complete ISP topology, is not feasible here because this study requires a trace to every BGP-advertised prefix (the unit of peering routing policy). Public traceroute servers cannot support this workload. However, PlanetLab, a distributed research test-bed, can.

As input for the analysis, I collected traceroute data from 42 PlanetLab [97] vantage points,[5] including sites in Australia, Canada, Denmark, Italy, New Zealand, Sweden, and the UK. From these vantage points, I ran traceroute to all of the 125,525 prefixes in the BGP routing tables of Route Views [80]. I used Scriptroute [116], a flexible network measurement facility, to collect the traces. Scriptroute helped speed up the trace collection process while reducing the network load with a few small optimizations: skipping name lookup, stopping when a loop is discovered, stopping after consecutive timeouts, and starting the trace a few hops into the network. It took only 6 hours to collect traces from most vantage points to all 125,525 prefixes.

I collected traceroutes on three consecutive days: December 18–20, 2002. Having three days of data was useful for two reasons. First, vantage points that fail during trace collection on one of the days contribute traces on another day to complete the picture. Second, datasets collected at different times help filter out transient paths that may be observed because of failures. Because the focus of this work was to study routing under stable conditions, I try to filter out the effects of instabilities. Previous research has shown that transient routing events generally do not last longer than a day [35, 74, 95].

6.3.3 Names of Locations

As in Section 4.3, I use DNS to find the boundaries of ISPs and to tell where routers are located. Using `undns` (Section 4.3.3), I reduce each IP-level traceroute to a POP-level path. Most traffic engineering decisions are made at the POP level [17, 124], which allowed me to focus on the most important effects. This step simplifies later analysis in three ways. First, it avoids the alias resolution process which I considered to be a potential source of error. Second, measuring a nearly complete backbone topology is much easier than obtaining a complete router-level topology. Inferring policy from an incomplete topology is much more difficult. Third, the POP-level topology has far fewer nodes (2,084 POPs compared to 52,000 addresses), simplifying the computation involved in routing policy inference and analysis.

6.3.4 Link Weights for POP-level Topologies

Inferred intra-domain routing policy makes it possible to predict which peering points represent early- and late-exits: the early-exit is the peering reachable with the least-cost path in the upstream ISP from where it enters the upstream; the late-exit is reachable with the least-cost path from where it leaves the downstream. To determine

[5] PlanetLab has since grown to over 120 sites.

Table 6.1: Tier-1 ISPs studied, organized by dominant regional presence. Many ISPs also have POPs outside their dominant operating region. "ASN" is the numeric identifier each ISP uses in BGP. "Degree" is the number of neighboring ASes as seen by RouteViews.

ASN	**Name**	**Tier**	**Dominant Presence**	**Degree**
4637	Hong Kong Telecom	1	Asia-Pacific	199
6453	Teleglobe	1	Canada	162
8220	Colt	1	Europe	161
3320	DTAG	1	Europe	111
3300	Eqip	1	Europe	67
7176	Genuity-europe	1	Europe	90
5511	Open Transit	1	Europe	172
1299	Telia	1	Europe	256
7018	ATT	1	US	1490
3561	Cable Wireless	1	US	806
1	Genuity	1	US	622
3549	Global Crossing	1	US	585
4513	Globix	1	US	455
3356	Level3	1	US	539
4006	Netrail	1	US	14
209	Qwest	1	US	887
1239	Sprint	1	US	1735
701	UUNet	1	US	2569
2914	Verio	1	US	538
7911	Williams Comm	1	US	234
2828	XO	1	US	184

Table 6.2: Tier-2 ISPs studied, organized by dominant regional presence. Many ISPs also have POPs outside their dominant operating region. “ASN” is the numeric identifier each ISP uses in BGP. “Degree” is the number of neighboring ASes as seen by RouteViews.

ASN	Name	Tier	Dominant Presence	Degree
2497	IIJ	2	Asia-Pacific	165
4725	One Data Network	2	Asia-Pacific	63
4755	VSNL	2	Asia-Pacific	49
9942	Comindico	2	Australia	114
15290	ATT-Canada	2	Canada	81
577	Bellnexxia	2	Canada	88
6539	Group Telecom	2	Canada	155
3602	Sprint-Canada	2	Canada	53
852	Telus	2	Canada	82
2686	ATT-EMEA	2	Europe	62
5400	Concert	2	Europe	117
4589	Easynet	2	Europe	86
13129	GATel	2	Europe	53
9070	ITDNet	2	Europe	20
174	PSI	2	Europe	46
3257	Tiscali	2	Europe	326
702	UUNet-europe	2	Europe	587
5669	Vianw	2	Europe	36
15412	Flag Tel	2	International	32
1668	AOL	2	US	156
7170	ATT-Disc	2	US	60
11537	Abilene	2	US	86
11608	Accretive	2	US	124
2548	Allegiance	2	US	216
1785	Appliedtheory	2	US	86
6395	Broadwing	2	US	231
16631	Cogent	2	US	187
4544	Conxion	2	US	43
5650	Eli	2	US	172
4565	Epoch	2	US	84
1784	Gnaps	2	US	63
6939	Hurricane Electric	2	US	42
10910	Internap	2	US	113
101	PNW-GPOP	2	US	28
7132	SWBell	2	US	112
4323	TWTelecom	2	US	277

Table 6.3: Tier-3 ISPs studied, organized by dominant regional presence. Many ISPs also have POPs outside their dominant operating region. “ASN” is the numeric identifier each ISP uses in BGP. “Degree” is the number of neighboring ASes as seen by RouteViews.

ASN	Name	Tier	Dominant Presence	Degree
2687	ATT-AP	3	Asia-Pacific	24
7543	Singapore Telecom	3	Asia-Pacific	8
1221	Telstra	3	Australia	66
6509	CANet	3	Canada	16
6467	Espire	3	US	30
3967	Exodus	3	US	43
6461	MFN	3	US	498
3701	Nero	3	US	12
4600	Oregon-GPOP	3	US	4
12050	TLCT	3	US	21
3582	University Oregon	3	US	5

which POP-level paths have least cost, I apply the router-level link-weight model developed in Section 6.1 with one small change.

The new problem when modeling intra-domain routing in support of peering routing policy analysis is that stub connections in the intra-domain topology have undefined link weights, illustrated in Figure 6.7. For intra-domain routing policy analysis alone, the inability to assign link weights to stub connections is not a problem: the path from A to I in the topology will be predicted correctly regardless of the value assigned to w_{gi}. However, if both H and I are nodes that provide peering connections to another ISP, it is important to compare w_{gh} and w_{gi} to find the early-exit when reaching destinations in that other ISP.

To solve this problem, I introduce a new constraint for each link that loosely ties the weight of the link to its latency, calculated using the geographic distance between points. The key observation is that operators often use latency as a component of link weight assignment, so using latency as a starting point in the analysis mirrors the practice of network operators.[6] One limitation of this analysis is that it will not model operator adjustment of IGP link weights on stub connections that are designed to modify the choice of early-exit. This limitation is because the inference of an early-exit peering routing policy and observations of early-exit paths do not provide feedback to the intra-domain policy inference to modify the link weight assignment. In the example of Figure 6.7, observing frequent use of node I as an early-exit peering point does not create a constraint like $w_{gi} < w_{gh}$.

The new constraint that associates link weight with latency is $w_{gh} - |e'_{gh}| = lat_{gh}$, where lat_{gh} is the latency of a link as suggested by geography and $|e'_{gh}|$ is a new error variable for the constraint solver to try to minimize. As the weighted sum of error values is minimized by the solver in the constraint hierarchy approach described in Section 6.1.3, these new error values have very low weight. The low weight of the new error values instructs the solver to keep link weights close to link latencies when it does not significantly affect the predictiveness of the rest of the intra-domain model.

I evaluated the predictiveness of the inferred weights as applied over the POP-level topology. I expected that, because traffic engineering decisions are made at the POP-level [17, 124], the traffic engineering decisions expressed at the router-level would manifest equally at the POP-level. In Figure 6.8, I compare inferred weights with a pure latency-based model in which the latency of an edge is used as its weight. This figure shows, for

[6] Link capacity is another starting point and link utilization may be used to refine the link weight setting; if these values were available, a better default weight model could be incorporated.

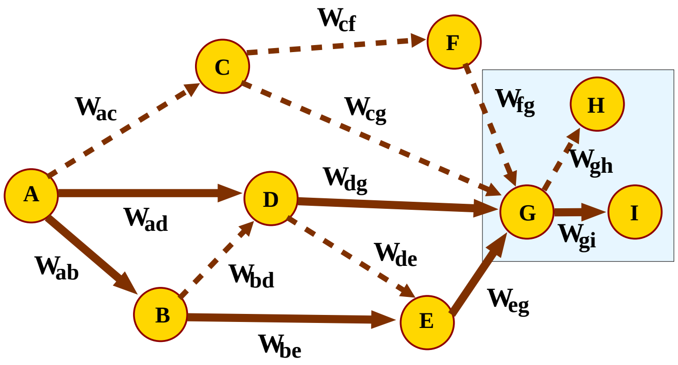


Figure 6.7: Stub connections in the topology have no alternate paths, and thus an undefined link weight. The shaded box encloses portions added to Figure 6.1. The link weights w_{gh} and w_{gi} can take any value and remain consistent with observed intra-domain routing. If there are peering points at H and I, however, correct assignment of w_{gi} and w_{gh} will support the prediction of which is the early-exit.

each ISP, what percentage of observed paths were least weight and what percentage were lowest latency. The ISPs are sorted based on the success of weights in describing their routing. For many ISPs, weights describe routing much more closely.

The inferred weights not only fit the observed paths well, they also predict paths between POPs for which no path was observed. I evaluated this by inferring weights from half of the dataset and analyzing how well these weights describe the complete dataset. To halve the dataset, I removed all the paths between a randomly-chosen half of the city pairs. Removing individual paths would only have reduced the count of observations. Figure 6.9 compares the success of the weights in describing the complete dataset when inferred using the complete dataset versus using only half of it. The weights inferred using half of the dataset predict routing nearly as well. I conclude that the inferred weights provide a concise and accurate description of POP-level ISP routing that is predictive beyond the paths present in the dataset used to infer it.

6.4 Peering Routing Policy Inference

In this section, I develop a technique to infer peering routing policy: a concise description of the relationship between pairs of ISPs as it influences the paths taken by packets. Each pair of connecting ISPs must decide: for packets crossing both networks, which ISP should do most of the work? This section focuses on how to determine what decision was made for every pair of connecting ISPs measured in Section 6.3.

The goal of peering routing policy inference is to annotate each directed edge in the inter-domain topology, the topology in which each node is an ISP as described in Section 2.1.1, with the type of high-level peering routing policy used. These high-level peering routing policies include early- and late-exit, defined in Section 2.2.2, among other variants as I will show. In contrast to intra-domain routing policy, in which each edge of the router-level topology is given a weight that represents a single configuration parameter of a router, peering routing policy associates with each inter-domain edge a summary of several configuration parameters that may be adjusted independently on every router in the ISPs. Although I will again try to provide one value for each edge, this value will instead summarize observed routing and thus summarize the many individual configuration elements that achieve that routing.

Peering routing policy inference has challenges like those of intra-domain routing policy inference. Foremost, the routing policy may not be universally applied. For example, MED values, those associated with destination prefixes to give a priority ordering to different peering points in support of late-exit, may not be

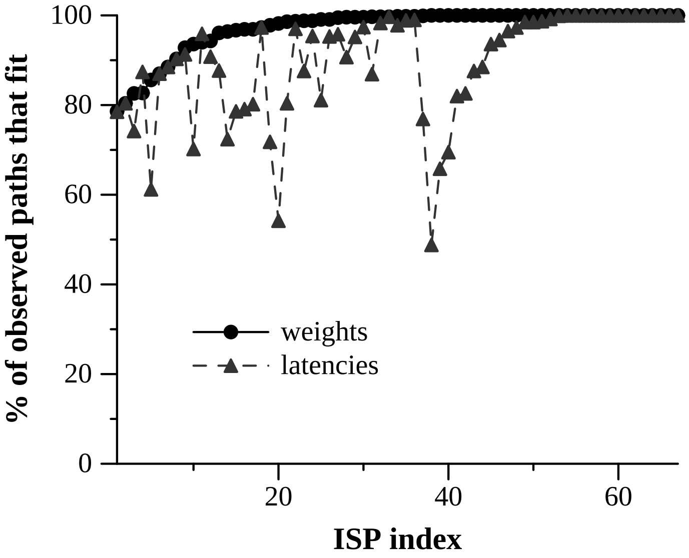


Figure 6.8: The success of the inferred weights compared to a pure latency model of link cost. Each point along the x-axis corresponds to an ISP. The y-axis values depict the percentage of observed paths that were least cost. This graph compares weights with latencies.

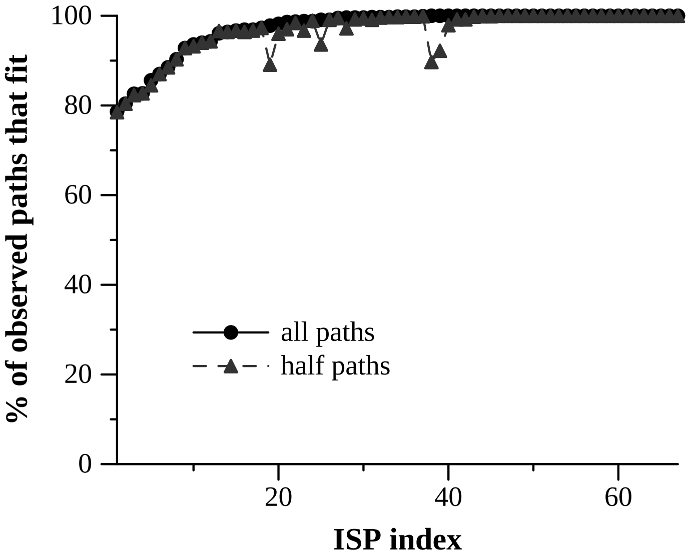


Figure 6.9: The success of the inferred weights in predicting paths not considered in the inference. Each point along the x-axis corresponds to an ISP. The y-axis values depict the percentage of observed paths that were least cost. This graph compares the weights inferred using the complete dataset with weights inferred using half of it.

associated with every prefix. Conversely, if the early-exit peering link for one source is congested, an ISP may choose to divert some of that link's traffic to another peering link. Again, this partial application of a routing policy can obscure the high-level routing policy that best summarizes observed behavior. Fundamentally, an observation of a single path carries less general meaning than in intra-domain routing policy because the underlying configuration parameters may be sensitive to the specific prefix for which the packet is destined, not simply where in the topology it is going.

Another challenge for peering routing policy inference comes from measurement errors like false or missing peering links. False links contribute paths that are not taken—when a false link appears to be the early-exit, it is rarely taken, suggesting that an early-exit routing policy is not in use. Missing links hide the alternate paths that show that a routing decision was made. The peering links most likely to be missed are those between ISPs using early-exit routing, but when only one peering is observed, one cannot be certain whether only one peering point exists (so no policy decisions need be made) or only one peering point was observed (presumably because of an early-exit policy).

Finally, to the extent that the ability to understand peering routing policy depends on an understanding of intra-domain routing policy, errors in intra-domain link weight inference will cascade.

6.4.1 Classifying Peering Routing Policies

To classify observed peering routing policies, I must determine whether observed paths are consistent with early-exit, consistent with late-exit, consistent with a partial application of both policies, or the result of a new policy. While early- and late-exit are defined by how close the peering point is to the endpoints, I observed an apparent load-balancing policy that was usually early, except when the early-exit peering point was congested. When the peering point was congested, another (possibly next-earliest, but not late) peering point was used for some of the traffic.

In Table 6.4, I show a small sample of the path data used to classify peering routing policy. A PlanetLab machine at Washington University in Saint Louis completed traceroutes to each of the 125,525 BGP-advertised prefixes in Route Views. Of these traceroutes, 93,918 traversed two adjacent ISPs in the chosen 65 ISPs. For each trace, the upstream and downstream AS number are recorded with ingress and egress POPs for each. Typically, the upstream egress and the downstream ingress are in the same city. From the tiny sample in Table 6.4 from just one vantage point, one might infer that only one peering point connects Verio to each of the other ISPs—in Chicago—and routing policy is not involved. If other peering points closer to the downstream egress are discovered by other vantage points, it is likely that early-exit is used from Verio to these other ISPs.

In Table 6.5, I show my list of peering routing policy classes, which I describe below.

Two methods can test whether observations are consistent with *early-exit* routing. One straightforward possibility is to test each observed path to see if the upstream egress is the peering to the downstream ISP having the lowest inferred intra-domain path cost. In Table 6.4, this rule would classify a routing policy as early-exit if the cost of the intra-domain path from St. Louis to Chicago is less than the cost of the intra-domain path to any other POP with a peering link to the downstream ISP. The problem with this method is that it is sensitive to error: false peering links (for example, a false peering in St. Louis) may cause the early-exit explanation to be discounted; and incorrect intra-domain link weight inference can misidentify the early-exit. Instead, a simpler classifier tolerates error: each ingress uses only one peering point for every destination reached through that AS. Small discrepancies can be tolerated with thresholds when defining "every" destination and "each" ingress. The example in Table 6.4 is consistent with early-exit for each of the ISP pairs because every time, a single peering (Chicago) is chosen. If the POP-level topology incorrectly discovers a peering in St. Louis, that it is not often used suggests it is not real. In BGP, anything is possible—it may be that the peering is real, but policy prevents its use for all but a select few sources and destinations—but the goal is a high-level classification that early-exit is used for most paths.

Table 6.4: Sample input to peering routing policy inference. These are 8 of 93,918 two-ISP paths observed by the PlanetLab site at Washington University in Saint Louis. Verio (2914) uses a peering point in Chicago to reach UUnet (701), AT&T (7018), Cable and Wireless (3561) and Global Crossing (3549). The path through each ISP is discarded. The downstream egress is not necessarily the location of the destination address; other ISPs may be involved in carrying the traffic between cities.

Destination	Upstream			Downstream		
Address	ASN	Ingress	Egress	ASN	Ingress	Egress
217.148.69.213	2914	St. Louis, MO	Chicago, IL	701	Chicago, IL	Barcelona, Spain
12.6.206.245	2914	St. Louis, MO	Chicago, IL	7018	Chicago, IL	Framingham, MA
200.69.44.155	2914	St. Louis, MO	Chicago, IL	3549	Chicago, IL	Miami, FL
204.249.220.199	2914	St. Louis, MO	Chicago, IL	701	Chicago, IL	New York, NY
211.29.67.179	2914	St. Louis, MO	Chicago, IL	3561	Chicago, IL	Los Angeles, CA
207.78.161.174	2914	St. Louis, MO	Chicago, IL	701	Chicago, IL	Chicago, IL
64.212.15.42	2914	St. Louis, MO	Chicago, IL	7018	Chicago, IL	Chicago, IL
216.77.189.118	2914	St. Louis, MO	Chicago, IL	701	Chicago, IL	New Orleans, LA

Table 6.5: Peering routing policy classes, briefly described.

Early-exit	Only one peering per ingress.
Engineered, not late	Several peering points per ingress; peering points are not closer in downstream link cost to downstream egress.
Late-exit, often	Many peering points per ingress; peering points are closer in downstream link cost to downstream egress.
Late-exit, sometimes	Late-exit, but for fewer than half of the paths.
Single	Only one peering point was observed for all ingresses.

Symmetrically, two methods can test whether observations are consistent with *late-exit* routing. The straightforward method is again to test for each observed path, whether the path left the upstream ISP at the point closest to the eventual destination in the downstream ISP, as calculated by intra-domain path cost. As with early-exit, this test is impractical because it is subject to measurement error. Instead, I classify late-exit by first, that more than one peering point is chosen from each ingress (that the peering routing policy is not early-exit), and second that each peering point is *closer* to the downstream egress than the early-exit, as determined by the IGP link weights in the downstream ISP.

These two categories are reasonable representations of intuition about what patterns of early- and late-exit routing should look like and are tolerant of measurement error; however, they are not prepared for the diversity of policies used in practice. The parameters used to configure BGP on each router provide operators with precise control over the routing of specific prefixes. It turns out that peering routing policy between a single pair of ISPs can be a mix of early- and late-exit. For this case, I create another category: *late-exit, sometimes*. This means that some late-exit routing is used between the pair of ISPs, but that if one had to guess for each packet of whether it would see an early-exit path or a late-exit path, the best guess would be "early." This distinction exposes a tension in this work: whether it is better to be predictive of path selection or be descriptive of the cooperative relationships between ISPs.

Another interesting behavior comes about when a pair of ISPs direct traffic in a way that is neither early—more than one peering is used per ingress—nor late—the peering is not closer to the downstream egress. I classify these as *engineered, but not late*. That is, operators appear to be modifying how packets are directed on a per-destination-prefix basis, but the observed paths are not consistent with late-exit because the observed peering points are not closer to the downstream egress. By "not closer," I literally mean further away than the early-exit peering. Finding such policies was unexpected, but an apparent consequence of under-provisioned peering links and a peering routing protocol that does not provide easy means of reducing load (as IGP link weights provide a means to reduce load by increasing the cost of a link).

The final class of policy is *single*. When only one peering point is seen between two ISPs, it is not possible to determine the peering routing policy: the single peering point is both the early- and late-exit. If some peering points were not observed, it is likely that the upstream ISP uses early-exit because at least one destination should have had a trace that discovered a late-exit peering point. Restated, to find all peering points in early-exit routing requires many sources while to find the peering points in late-exit routing requires many destinations; this mapping used 42 sources and 125,525 destinations, so missing peering points in early-exit is more likely. Figure 6.10 illustrates the effect of early- and late-exit peering policies on the discovery of peering points.

"Single" is a separate class even though that the underlying configuration is probably consistent with early-exit both when the measurement is correct (no more peering points exist) and when the measurement is not (more peering points exist, but were missed because the policy was early-exit). What I will study with these classifications is the asymmetry in relationships between ISPs of different sizes. A small ISP with a single connection to its upstream provider should not be counted in such an analysis, so the separate category for "single" is useful.

I describe the results of these classifications in Chapter 7, but evaluate their predictive power below.

6.5 Peering Routing Model Predictiveness

The goal of this section is to determine whether the inferred peering routing policy is predictive of paths through ISP pairs. The objective is to predict, given the POP-level endpoints of a path crossing two ISPs, which peering point will be used: where a packet will leave the upstream ISP for the downstream.

I intended to use inferred peering routing policy to characterize the cooperation between ISPs and expose the diversity of peering routing policies. Evaluating whether the model is predictive of path selection provides a hint of whether further study is warranted: how much more detail may be needed to predict more accurately.

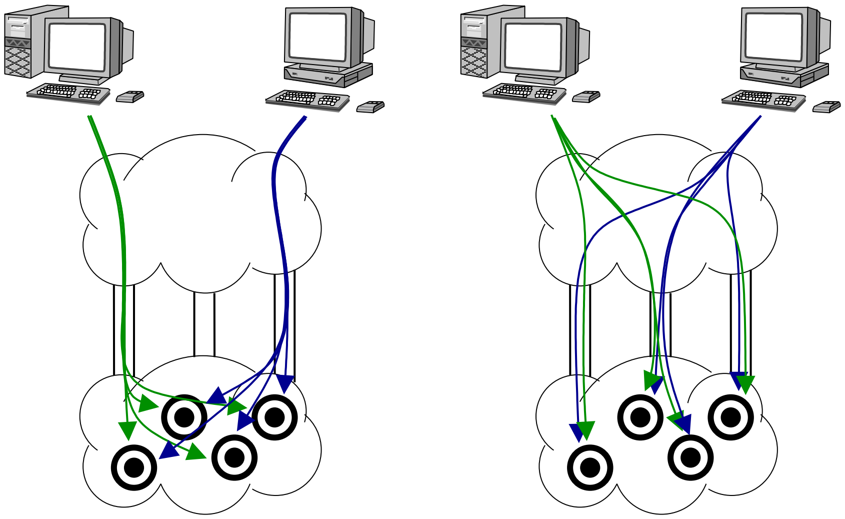


Figure 6.10: The effect of different peering policies on the completeness of peering point discovery. Clouds represent ISPs, targets represent destinations, and workstations represent PlanetLab vantage points. Early-exit (left) stresses the diversity of sources; if the sources are not sufficiently diverse, early-exit may hide peering points, such as the one in the middle. Late-exit (right) stresses the diversity of destinations, which is easier to increase. Early-exit routing and insufficient vantage points may combine to mis-classify ISP pairs as "single."

Such detail might include a list of peering links that are load-balanced or a list of prefixes routed using late-exit. For this work, however, prediction accuracy is not the objective.

In this section, I quantify how well the inferred peering routing policies can be used to predict which peering link will be used. The concrete goal is to fill in a table like Table 6.6. When working on the first row of this table, suppose the ISP pair 2914:701 (Verio to UUnet) has been classified as following an early-exit routing policy. One step remains: to determine the early-exit peering point for Verio in St. Louis to reach UUnet. I compare two methods. Recall from Section 6.4.1 that early-exit is classified based on the use of a single, dominant peering point for reaching the vast majority of prefixes through the downstream ISP. This dominant peering seen provides one approach: include an explicit mapping from ingress POP to early-exit peering point for each ISP pair. A more compact solution is to rely on the inferred IGP link weights to determine, given a list of peering points, which peering point is nearest.

If a pair is instead classified as "late-exit, often," the late exit must be predicted. Unfortunately, the dominant path technique would require storing a great deal of information—policy is applied not to the ingress POP but to the destination prefix, and destination prefixes vastly outnumber POPs. I use only minimum IGP link cost to determine the late exit.

All other classifications are predicted as if they were routed using early-exit, which is their dominant approach, and appears to be the best guess with the information available.

As for intra-domain routing in Section 6.2, the accuracy of the routing model can be tabulated in two ways: by observed path and by dominant path.[7] Accuracy by observed path is an estimate of how likely the model is to predict the peering point correctly for any randomly chosen destination prefix. Accuracy by dominant path, where all observations between the entry to the upstream ISP and exit of the downstream ISP are collapsed, is less biased by the number of times a path was observed. I consider a dominant path to be accurately modeled if

[7] The third method, which evaluates whether all chosen paths are observed, is not needed here because, to my knowledge, equal-cost multi-path routing is not a feature of BGP.

Table 6.6: Using the paths observed in Table 6.4, the predictiveness of a peering routing model is how well it can predict the correct peering point (upstream egress) given the upstream ingress, downstream egress, and ISP pair. The destination address and downstream ingress are not considered here.

Upstream			Downstream	
ASN	Ingress	Egress	ASN	Egress
2914	St. Louis, MO	?	701	Barcelona, Spain
2914	St. Louis, MO	?	7018	Framingham, MA
2914	St. Louis, MO	?	3549	Miami, FL
2914	St. Louis, MO	?	701	New York, NY
2914	St. Louis, MO	?	3561	Los Angeles, CA
2914	St. Louis, MO	?	701	Chicago, IL
2914	St. Louis, MO	?	7018	Chicago, IL
2914	St. Louis, MO	?	701	New Orleans, LA

95% or more of the paths between that node pair are correctly predicted. I chose this threshold to ignore noise; results with 90% and 99% were similar.

Using the IGP link weights to determine the early-exit peering point achieves a prediction accuracy by observed path of 76%. Using the dominant chosen peering point instead can reach 87%. By dominant path, the prediction accuracy of the two approaches are much more similar: the dominant peering point accuracy is 89%, while the IGP link weight model correctly predicts a slightly smaller 88%. That the IGP link weight model does much better across dominant paths may be because of a few, very popular paths that are mis-predicted by the IGP link weight model. The IGP link weight model will predict incorrectly if a false peering link is detected or if the intra-domain link cost is not inferred correctly as in Figure 6.7.

The accuracy of these measures does not exceed 90%. This accuracy is less than that of intra-domain routing policy because peering point selection is complicated and special cases are common. In Chapter 7, I will show the diversity of peering point policies and that different approaches, such as late-exit, may only be applied in certain places or for certain prefixes. These ad-hoc tweaks to routing policy do not lend themselves to simple modeling.

6.6 Summary

The inference of routing policy described in this chapter is based on the key insight that paths *not* taken expose the routing policy choices made by ISP operators. The measured ISP network maps include many alternate paths, which can be used to develop a simple summary of routing policy. For intra-domain routing policy, this summary is a list of estimated edge weights. For peering routing policy, this summary is a classification as to whether the peering selection is early-exit, late-exit, or somewhere in between.

I evaluated the accuracy of the inferred routing policies by measuring their prediction accuracy: the fraction of paths accurately predicted. For most ISPs I studied, intra-domain routing policy inference generated a solution that predicted over 90% of the paths correctly. Peering routing policy inference, though given only a few relatively coarse options, only predicted 88% of paths correctly. Unfortunately, the path chosen for one destination provides less information in BGP than it does in an IGP: BGP configuration, such as localpref and MEDs, can be applied differently for each prefix and at each router, while IGP configuration is regular and assigns a single value to each link.

Inferred routing policies are valuable for developing an understanding of how the topology is configured. The ability to compare link weights to geographic latency provides a view of how well-provisioned the network is: deviation from the geographically shortest path suggests that some of its links are under-provisioned. In peering routing policy inference, late-exit or other special routing policies suggest, again, under-provisioned links or a special cooperative relationship between pairs of ISPs.

Chapter 7

ISP MAPS AND ANALYSES

In this chapter, I present and analyze the ISP maps measured using my techniques. These maps provide the most comprehensive quantitative view of ISP topologies and routing policies measured by an outsider.

The first section of this chapter presents both sample backbone and POP maps to show the heterogeneity of backbone designs and a representative picture of the topology of a small, measured POP.

The second section of this chapter presents an analysis of the Rocketfuel maps. The focus of this section is understanding the structure of the maps, taking advantage of geographic and role information to understand how one might build a structural model of ISP networks and stitch structural models of different ISPs together into a synthetic Internet topology. In particular, I study router degree distributions, the fraction of backbone routers in POPs, and how many POPs have peering links to neighboring ASes.

The third section of this chapter considers the inferred routing policies. I use inferred intra-domain routing policy to understand the circuitousness of network paths, which provides an indirect measure of the provisioning of ISP networks. A goal was to look for evidence of congestion within ISP networks by observing how operators might route around congestion. I use peering policies as a means to understand how ISPs cooperate and compete. For example, a late-exit peering routing policy requires special configuration to implement, so suggests there is an interesting relationship between two ISPs: the upstream ISP that carries packets further than otherwise necessary is perhaps being paid to do so. Or, perhaps the goal of the upstream ISP is to keep its customers' traffic off a lower-quality downstream ISP's backbone. Although observed routing policy does not explain why a specific configuration is used, it points to the heterogeneity of relationships between large ISPs.

7.1 Measured Router-level ISP Maps

In this section, I present a sample of measured backbone topologies of ISPs in the United States and a measured topology of routers in a POP. This section is the first of two that characterize the Rocketfuel dataset measured in Chapter 4 and evaluated in Chapter 5. This section includes two lessons: backbone designs vary and POP designs are largely similar. The figures in this section will provide visual context for the more detailed analysis that follows in the next.

7.1.1 Backbone Topologies

Figure 7.1 shows five measured ISP backbones overlaid on a map of the United States. A key result of my work is that backbone design style varies widely between ISPs. AT&T's backbone network topology includes hubs in major cities and spokes that fan out to smaller per-city satellite POPs. In contrast, Sprint's network has only 20 POPs in the USA, all in major cities and well connected to each other.

Level3 represents yet another paradigm in backbone design: the mesh is most likely the result of using an underlying circuit technology, such as MPLS, ATM, or frame relay PVCs, to tunnel between POPs. Recall from Section 3.1.3 that traceroute sees only the logical, IP-level topology, which does not always represent physical connectivity. The properties of the underlying network connectivity in Level3's network are unknown, and as such, I will often remove Level3 before studying aggregate properties of all ISPs.

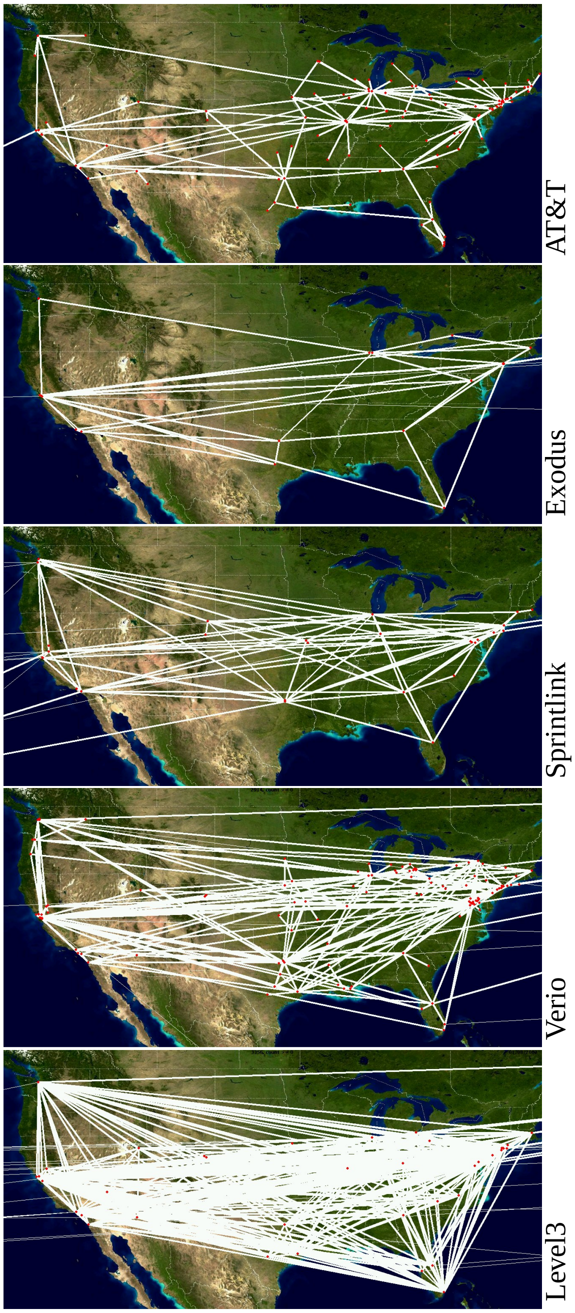


Figure 7.1: Measured backbone topologies of US ISPs with inferred geographic location from DNS names. From top to bottom: AT&T, Exodus, Sprint, Verio, and Level 3. Multiple links may be present between two cities; for clarity, I show only one. The background image is from NASA's visible earth project.

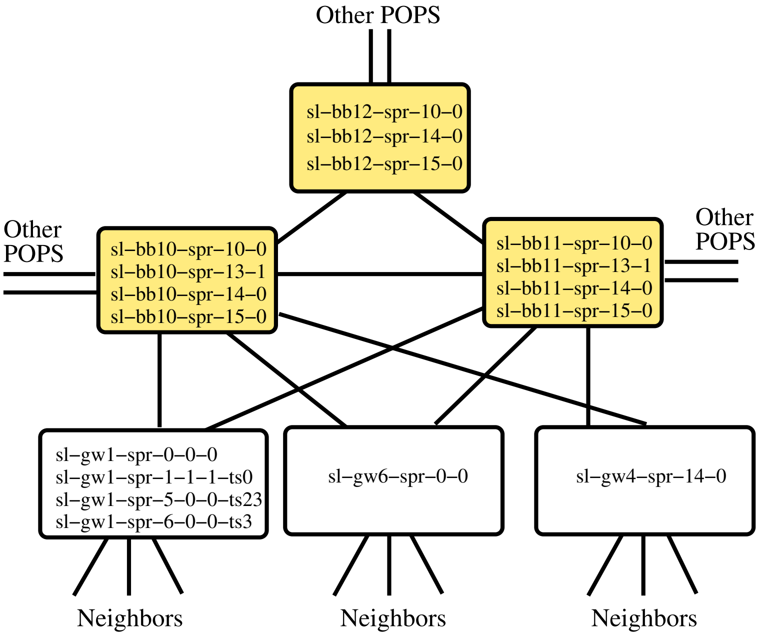


Figure 7.2: A sample POP topology from Sprint in Springfield, Massachusetts. Shaded boxes represent backbone routers. The names are prefixes of the full names, without the common sprintlink.net suffix. Interfaces on the same router usually have different names; these aliases are listed in the same box. Most POPs in Sprint are larger and too complex to show, but have a similar structure.

7.1.2 Router-level Topologies of POPs

Unlike the backbone designs, POP designs are relatively similar. Recall from Section 2.1.2 that each POP is a physical location where an ISP houses a collection of routers. A generic POP has a few backbone routers that connect to each other in a densely-connected mesh. In large POPs, backbone routers may not be connected in a full mesh. Backbone routers also connect to backbone routers in other POPs. Each access router connects to routers from neighboring ISPs (or customers) and to at least two backbone routers for redundancy, but does not connect outside the POP. Some neighboring routers may not use a point-to-point link to connect to an access router. Instead, a layer 2 device such as a bridge, or a multi-access medium such as a LAN may aggregate neighboring routers that connect to an access router. A limitation of this study is that traceroute cannot differentiate these layer 2 topologies from point-to-point connections.

As an example of a common pattern, Figure 7.2 shows a measured map of Sprint's POP in Springfield, MA. This POP is small; large POPs are too complex to show here in detail. In the figure, names of the aliases are listed together in the same box. The three backbone nodes are shown on top, with the access routers below. Sprint's naming convention is apparent: sl-bb$n$ names backbone routers, and sl-gw$n$ names their access routers. The value of DNS names for understanding the role of routers in the topology is clear from this naming practice.

This hierarchical pattern is common, but as I show in Section 7.2.2, the number of backbone routers needed varies widely, both for a given POP size and for a given number of connections to other POPs.

7.2 Analysis of the Measured Router-level ISP Topologies

In this section, I analyze the router-level ISP topologies measured using the Rocketfuel techniques presented in Chapter 4 and evaluated in Chapter 5. I first describe the router out-degree distribution, then the composition of POPs, the degree distribution of POPs, and finally the number of external (peering) connections at each POP. Each of these properties can be useful for synthetic generation of ISP topologies and helpful in understanding how the Internet is assembled from ISPs.

This analysis characterizes attributes of structured ISP network topologies that would be useful in the synthetic generation of ISP topologies and in understanding how widely ISPs in the Internet connect. POPs and ISPs are fundamental components of the structure of the Internet, and the analyses of this section are intended to enable synthesis and assembly of these fundamental structural components. While I do not provide a new topology generator, the structured topologies I measured in this work provide a new source of input for how one might synthetically generate an Internet topology.

7.2.1 Router Degree Distribution

The out-degree of a router is how many neighbors it has. The distribution of node out-degree is one popular [22, 40, 55] method of characterizing the topology of a network. When this work began, power-law degree distributions had recently been found in network topologies [40] including the early router-level topology measured by Pansiot and Grad [92]. With the same number of nodes and edges, a graph with a power-law degree distribution is more likely to feature short paths and redundancy than the more traditional Erdös-Rényi random graph [39] that formed the foundation of early synthetic models of Internet topology such as the Waxman model [132]. The observation of power laws spurred research into understanding how they come about [10, 29, 41], whether the network is more robust because of them [2], whether their observation might be the result of how maps were measured [69], how to generate synthetic topologies with the same properties [23, 79], and whether the degree distribution of router topologies is better modeled by a different distribution such as the Weibull [22]. Because Rocketfuel maps include information about structure and have more detail in individual ISPs, the maps I measured contribute new information to this discussion: which routers have such high degree and whether all ISPs have similar degree distribution. I was particularly curious whether power laws might only appear in whole-Internet maps, in which the individual designs of different ISPs blur together.

To graph the distribution of router out-degree in the ISP networks I use the complementary cumulative distribution function (CCDF). A CCDF plots the probability (y-axis) that the observed values are greater than the ordinate (x-axis). This style of graph pulls out both features of power-law distributions: that there are a few items (nodes) having very high value (degree), which appear at the bottom-right corner of the graph, and many items having very low value at the top-right corner. A true power-law distribution will appear to be a straight line on a log-log CCDF plot.

Recall from Section 5.7 that although the maps are reasonably accurate, the using traceroute from finite (though many) vantage points yields a sampling bias that may induce the appearance of a power-law degree distribution where it does not exist. This measurement is not free of sampling bias.

I first show the degree distribution of all routers, regardless of their role in the ISP, then focus on just the backbone routers.

7.2.1.1 Degree distribution of all routers

Figure 7.3 shows the CCDF of router out-degree in the aggregate over all ISPs and Figure 7.4 for individual ISPs. The tails of these distributions are fit using Pareto (“power-law”), Weibull, and lognormal distributions. The α parameter for the Pareto fit is estimated over the right half of the graph to focus on the tail of the distribution. The Weibull scale and shape parameters are estimated using a linear regression over a Weibull plot. The lognormal line is based on the mean μ and variance of the log of the distribution.

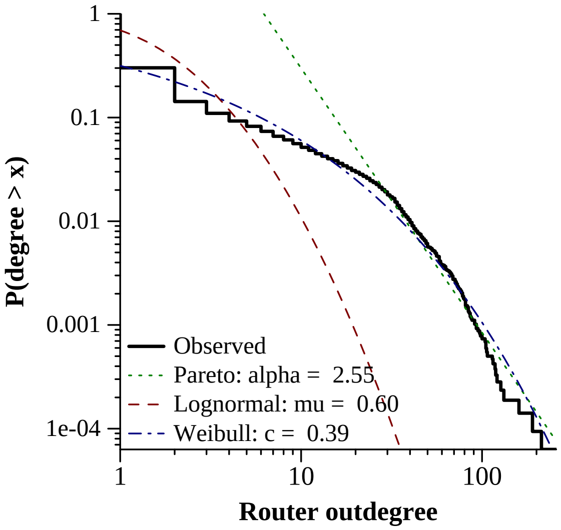


Figure 7.3: Router out-degree CCDF. The Pareto fit is only applied to the tail. The mean out-degree is 3.0; 65% of routers have a single link within the ISP. This graph is an aggregate over nine of the ISPs; I exclude Level3 because of its logical mesh topology.

Unlike the measured degree in AS graphs [40], router out-degree has a small range—it covers only two orders of magnitude over the ten ISPs. Physical size and power constraints limit the number of interfaces on each router, and in turn, the underlying router out-degree. The number of physical interfaces does not always limit the router out-degree: the router-level topology can include undetected layer two switches and multi-access links, which would make the observed router out-degree much larger than the number of physical interfaces.

Different distributions fit different ISPs best in Figure 7.4: some are fit by Weibull, a few by the simpler lognormal, and most ISPs I studied have tails that are consistent with the Pareto fit. Weibull fits both the tail and body of many of the distributions. Although these distributions have significant tails, the α parameter is high for a heavy-tailed distribution. *Although the distribution of router out-degree is skewed and highly variable over its small range, the out-degree distribution within these measured ISP topologies is not, technically, heavy tailed.*

The absence of a heavy-tailed distribution in ISP router-level degree distributions has two main implications. First, researchers should not use power-law-based topology generators to construct ISP network topologies. They may remain appropriate for whole-Internet maps: I do not show that the whole-Internet lacks such a degree distribution, only that the ISPs I studied, individually and in aggregate, do not show power laws. Second, if the appearance of power laws is a result of sampling bias, the absence of power laws in these measurements strongly suggests that power laws are not real features of ISP topologies. That I find no true power laws in these topologies does not, however, invalidate results based on the underlying, highly-variable, positively-skewed distribution.

7.2.1.2 Degree distribution of backbone routers

After observing the degree distributions of all routers, I focused on the degree distribution of routers classified as “backbone” routers. Backbone routers are classified as such by their connections to other POPs, though this may not match the ISP’s own concept of a backbone. I define backbone in this ISP-independent way because DNS tags that represent the ISP’s idea of a router’s role in the topology are not universally used. The question

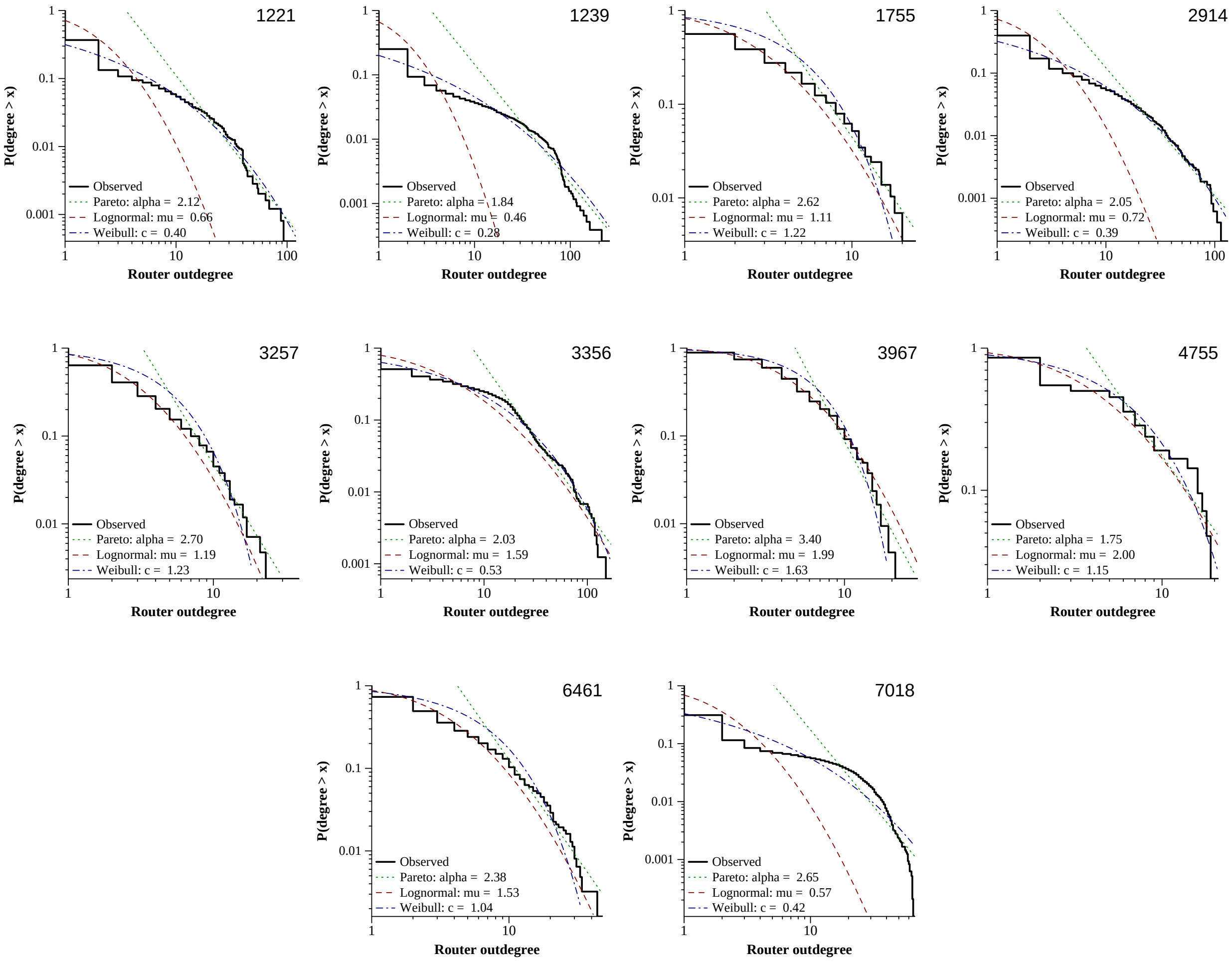


Figure 7.4: Router out-degree CCDF by ISP. The names of these ISPs can be found in Table 5.1 on page 35.

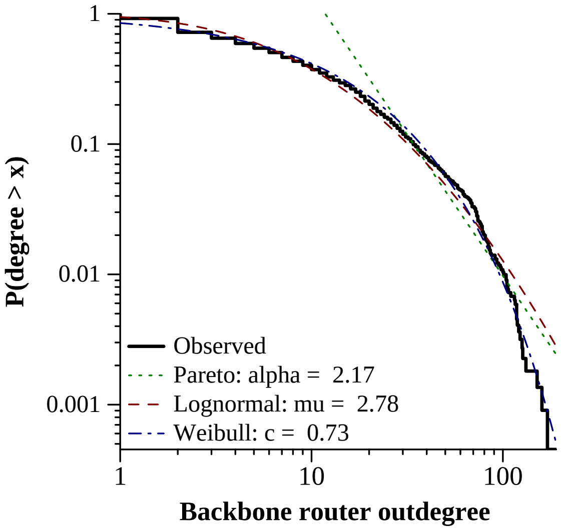


Figure 7.5: Backbone router out-degree CCDF. The Pareto fit is only applied to the tail. The mean out-degree is 11.7, the median is 5. This graph is an aggregate over nine of the ISPs; I exclude Level3 because of its logical mesh topology.

is whether the different roles of routers in the ISP topology are exposed through different connectivity in the degree distribution.

The same out-degree analysis over only backbone routers produces a visually different distribution in Figure 7.5. This distribution of backbone router out-degree visually follows the lognormal curve. While most ISP routers are “leaves” in that they connect to only one other ISP router, (over 65% as shown in Figure 7.3) most backbone routers have high out-degree. I conclude that *the backbone routers serve a noticeably different purpose in the topology—providing rich connectivity.* Other routers in the network, while they may connect widely externally, appear more likely to act as stubs having limited connectivity within the ISP network. Restated, router role and router out-degree appear related: the structure of the ISP network is important.

7.2.2 POP Composition

Given that strict adherence to the degree distribution does not appear to capture the structure of ISP networks, I now focus on understanding how the larger components of Internet topology (POPs, ISPs) might be assembled in a structural model. The first element I consider is the distribution of POP sizes and how routers are distributed across these POPs. I then consider the role of backbone routers in these POPs: how many are present by POP size and by how many connections there are to other POPs.

7.2.2.1 POP sizes

Figure 7.6 shows the distribution of measured POP sizes, aggregated over the ten ISPs I studied. Most POPs have fewer than four ISP routers, but most routers are in POPs with more than sixteen. Small POPs may be called by other names within the ISP; I do not distinguish between exchange points, data centers, and private peering points. *The distribution of POP sizes is positively-, or right-, skewed, meaning most POPs are small, but most routers are in relatively large POPs.*

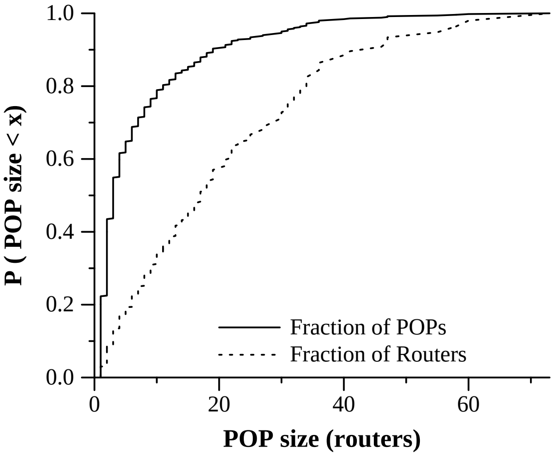


Figure 7.6: The cumulative distribution of POP sizes (solid), and the distribution of routers in POPs of different sizes (dotted). The mean POP size is 7.4 routers, and the median is 3 routers.

7.2.2.2 Router roles in POPs

To stitch together synthetically-constructed POPs, one needs to know which of the routers in the POP are backbone routers that connect to other POPs and which connect only within the POP or to customers. Based in part on the measured POP-level topology in Figure 7.2, I expected to find that the number of backbone routers in a POP would scale sub-linearly with the total number of routers in that POP. That is, that larger POPs would have backbone routers with more interfaces. What I found instead was that *larger POPs have more backbone routers*, commensurate with the number of total routers in that POP.

In Figure 7.7, I show the number of backbone routers relative to the total number of routers in the POP, for the ISPs I studied. “Backbone” routers connect to other POPs, and I include only those routers identifiable by DNS name and IP address as being part of the ISP. Unsurprisingly, most of the routers in small POPs are used to connect to other POPs, likely to the better connected core of the network. Although I expected that as POPs became larger, a smaller fraction of backbone routers would be required, POPs with more than 16 routers vary widely in the number of backbone routers used to serve them. This graph shows that the smallest POPs have multiple backbone routers for redundancy and larger POPs vary widely in the number of backbone routers present.

If the number of backbone routers varies unpredictably with POP size, perhaps it varies more simply with POP-degree: the number of other POPs to which this POP connects. This hypothesis is based again on the assumption that backbone routers, when connecting to many other “things” (access routers above, now other POPs), can scale using interfaces instead of more backbone routers.

In Figure 7.8, I show the out-degree of a POP as a function of the number of backbone routers present. I was surprised to find a roughly linear relationship, having similar variability as the relationship between backbone routers and POP size. The median tracks a line where the out-degree of a POP is equal to the number of backbone routers present. However, in some POPs, only one or two backbone routers connect to several other POPs. Conversely, some POPs have several backbone routers that provide redundancy in connecting to a just a few other POPs. I conclude that *there is no standard template for how backbone routers are connected to other POPs.*

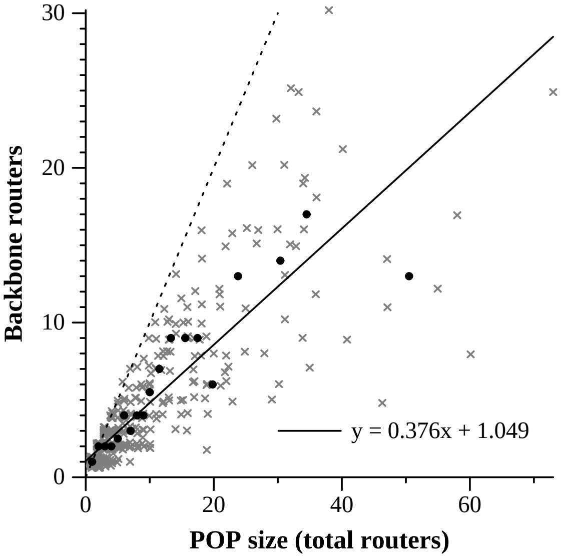


Figure 7.7: Backbone routers in a POP relative to its size. A small random jitter was added to the data points to expose their density. Circles represent the median of at least ten nearby values: fewer medians are present for the few large POPs. The dotted line follows $x = y$, where all routers in a POP are backbone routers. The solid line traces a linear regression fit with $R^2 = 0.69$. This graph aggregates data from the ten ISPs.

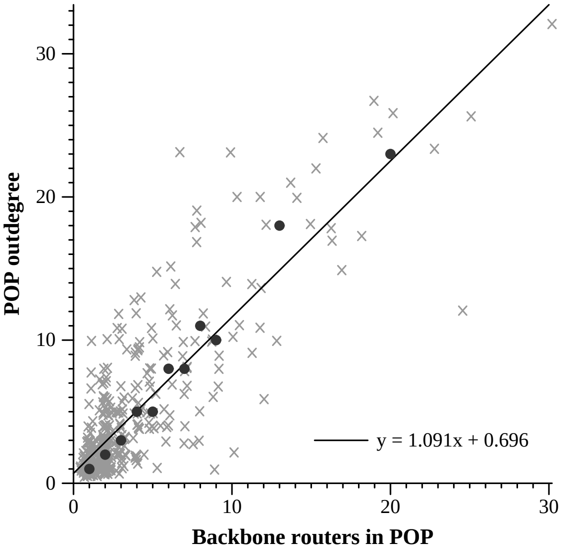


Figure 7.8: POP out-degree vs. backbone routers in the POP. A small random jitter was added to the data points to expose their density. Circles represent the median of at least ten nearby values: fewer medians are present for the few large POPs. The solid line traces a linear regression fit, with $R^2 = 0.70$. This graph aggregates data from nine ISPs; I exclude Level3 because of its logical mesh topology that gives POPs very high out-degree.

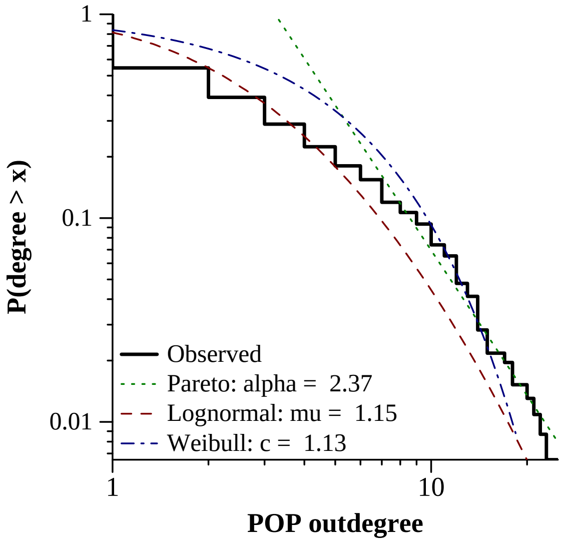


Figure 7.9: POP out-degree CCDF, which represents the node degree distribution over the backbone graph where each node is a POP. The mean out-degree is 3.5, the median out-degree is 2. This graph aggregates data from nine of the ISPs: I exclude Level3 because of its logical mesh topology.

7.2.3 POP Degree Distribution

I now look at the coarser, POP-level topology found in the Rocketfuel ISP maps. This topology is represented by the backbone graph: POPs are the nodes and bidirectional backbone links connect them. Multiple links between POPs are collapsed into a single link. Figure 7.9 shows the POP out-degree distribution. This distribution is similar to that of the router-level topology, though over a smaller range. Along the left-edge of the graph, nearly half of the POPs are stubs that connect to only one other POP. On the right side of the graph, several POPs act as hubs. Figure 7.9 does not include Level3: the layer-2 switched architecture it uses creates a large mode at backbone out-degree around 50.

7.2.4 Peering Structure

The Rocketfuel maps were collected using traceroutes that enter and exit the ten ISPs at diverse points, providing the unique opportunity to study the link-level peering structure connecting ISPs. Adjacencies in BGP tables show only that pairs of ASes connect. Rocketfuel topologies include where and in how many places the ISPs I studied exchange traffic. For example, while BGP tables show that Sprint and AT&T peer, the Rocketfuel topologies show where the two ISPs exchange traffic. Understanding this relationship provides some of the “glue” needed to stitch together ISP topologies to form a synthetic, structural whole-Internet map.

I summarize the link-level peering structure by showing the number of locations where the mapped ISP exchanges traffic with neighboring ASes. Neighbor ASes may be other ISPs, whether in a transit or peer relationship, or customers running BGP. I use the same complementary cumulative distribution function (CCDF) plot style to show the distribution. Figure 7.10 plots this CCDF, aggregated over the ISPs I studied. The Pareto, lognormal and Weibull fits are calculated as in Section 7.2.1.

The distributions are highly skewed for each of the ISPs I studied individually and in the aggregate. *Each ISP is likely to peer widely with a few other ISPs, and to peer in only a few places with many other ISPs.*

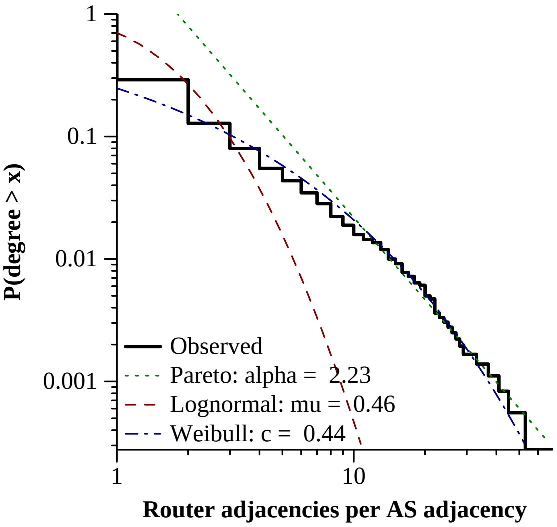


Figure 7.10: A CCDF of the number of router-level adjacencies seen for each AS-level adjacency. AS-level adjacencies include connections to other ISPs and connections to customers that manage their own AS.

These relationships are perhaps not surprising given that the distribution of AS size and AS degree are heavy tailed [29].

The number of router-level adjacencies for each AS adjacency varies over a small range, covering only one to two orders of magnitude. Some of the "peers" with many router-level adjacencies are different ASes within the same organization: AS 7018 peers with AS 2386 in 69 locations and with AS 5074 in 45 locations, but all three represent AT&T. Discounting these outliers, the graphs show that *it is rare for ISPs to peer in more than thirty locations.*

The next question is whether peering is concentrated in only a few POPs, such as those in populated cities. In Figure 7.11, I show a CCDF of the number of peering connections per POP. This graph relates to the out-degree graphs previously presented in that this shows the out-degree of a POP in the number of its external connections. *The ISPs I studied connect to hundreds of other ASes in a few central cities.* However, most cities house only a few external connections.

7.3 Analysis of Routing Policies

In this section, I study the two lower layers of network routing policy: intra-domain and peering. (Inter-domain routing policies have been somewhat well studied through analysis of Route Views data.) I apply the POP-level measured maps of Section 6.3 to characterize both routing policies: these maps provide an understanding of the internal network engineering and pairwise peering policies of 65 diverse ISPs, a dataset much larger (and potentially more representative) than that of Rocketfuel. Although these network maps are at the POP-level, and thus less detailed than those of the ten Rocketfuel ISPs, they are complete enough to study patterns of routing policy.

In this section, I first present a characterization of intra-domain routing policies, showing that they are consistent with shortest-latency paths and thus consistent with adequate provisioning within ISPs. I then characterize peering routing policies by the prevalence of early- and late-exit and find heterogeneity and asymmetry in the routing policy choices made by different ISPs.

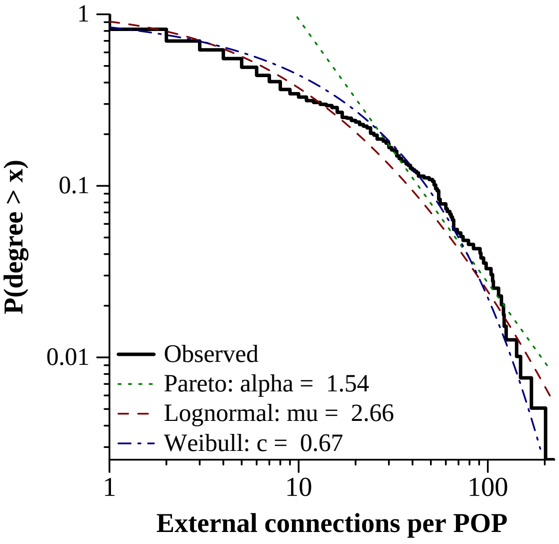


Figure 7.11: A CCDF of the number of external adjacencies per POP. Some POPs are particularly important, while most have at least a few external connections.

7.3.1 Intra-domain Routing Policies

Investigating intra-domain routing policy is a means to understand the design goals of ISP networks. Here, I will use the absence of circuitous paths as evidence that ISP networks are adequately provisioned. *Adequately provisioned* means that each link within an ISP has the capacity to carry traffic along shortest-latency paths. If certain links existed in the topology but were over-used (under-provisioned), one would expect ISP operators to use routing policy to divert traffic away from the congested part of the network. Determining whether this occurs is the subject of this section.

Detecting circuitous routing within ISPs is not simple. To try to simplify the presentation, I use the concept of *path inflation*: the extra length of the path through the topology beyond the “shortest” path. Path inflation has been used as a measure of routing-protocol effects by Gao and Wang [50] and Tangmunarunkit, *et al.* [125, 126]. I report path inflation in units of time (milliseconds) because it is a more accessible measurement of performance, although the underlying calculation is based in the geographic distance between POPs (rather than a direct measurement of link latency).

Across the POP-level topologies of the 65 ISPs studied in Section 6.3, I calculate the path inflation caused by topology and by policy. The path inflation caused by the topology is calculated, for every pair of POPs, as the difference between the distance traveled through the shortest path of POPs and the direct geographic distance between the two POPs. That is, path inflation from topology is the difference in length between the shortest path and a hypothetical direct link. The path inflation caused by routing policy is calculated, for every pair of POPs, as the difference in length between the path chosen by intra-domain routing policy and the shortest-latency path.

I show both these measures of path inflation because path inflation from topology provides context. If the topology did not support direct paths, whether the routing policy chooses the most direct path available seems less important. On the other hand, if the topology includes many direct paths but these direct paths are not used, routing policy is important and can be used as a means to identify congestion.

I exclude ISPs that appear to use virtual circuit technologies such as MPLS. This prevents underestimating path inflation, as these topologies have many IP-level edges without a corresponding physical link. I identified six such networks by their unusually high ratio of edges to nodes. I also exclude city pairs in the ISP that were

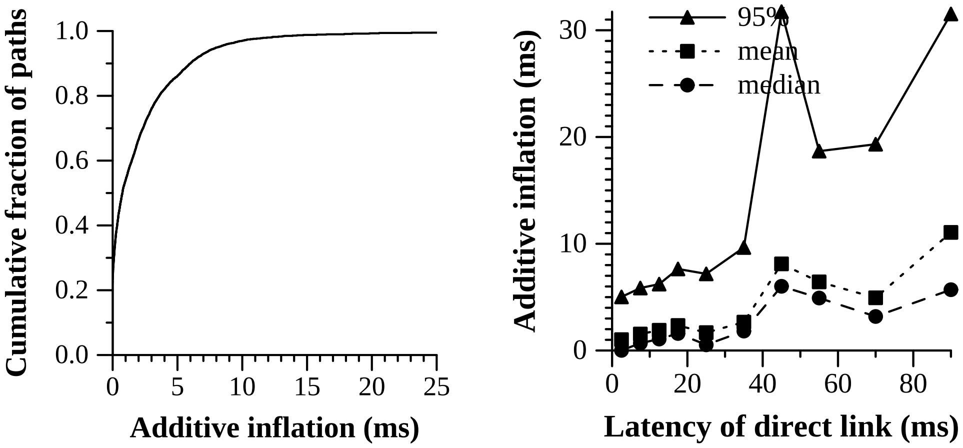


Figure 7.12: Path inflation due to intra-domain topological constraints. The left graph shows the CDF of path inflation, and the right one shows the median and 95th percentile inflation as a function of latency of a hypothetical direct link.

never observed in the same trace. This prevents overestimating the inflation caused by topology if the traces missed a link that would provide a shorter path between the cities.

7.3.1.1 Inflation from intra-domain topology

The graph on the left of Figure 7.12 shows the cumulative distribution function (CDF) of path inflation. Most paths are not inflated by much, pointing to the well-connectedness of most ISPs. The median, mean, and 95th percentile inflation were 2, 3, and 10 ms. As I explain below, the high inflation points correspond to city pairs on different continents.

The graph on the right characterizes inflation as a function of the reference distance between city pairs. I place each POP pair into a bin by the direct geographic distance between them. Distances less than 20 ms, expressed in the time for light to travel through fiber, are mapped to 5 ms bins (0–5, 5–10, 10–15, and 15–20); distances between 20 and 60 ms are mapped to 10 ms bins (20–30, 30–40, 40–50, 50–60); and longer distances are mapped to 20 ms bins. This binning helps study the variation of inflation with the direct distance and provides a sense of relative inflation: whether it is the short or the long paths that are made longer. The results are insensitive to bin sizes when the bins are small enough to expose interesting effects yet large enough to have enough data points for meaningful statistical measures.

The graph plots the median, mean, and 95th percentile of data points in each bin. Paths between most pairs of geographically close cities are inflated only by a small amount, indicating that these cities are usually connected by short paths. The jump around 40 ms represents cities that are on different continents; such cities are forced to reach each other by one of a few inter-continental links. The right end of the graph represents city pairs in Europe and Asia/Australia that have to traverse the USA. Because I removed city pairs that were not seen together in a trace, this jump is not a consequence of most vantage points being in the USA—the data contained intra-domain paths from European vantage points to destinations in Asia via the USA.

Path inflation due to topology in the tier-1 ISPs (4 ms) was higher than in other ISPs (2 ms). This difference is caused by the tier-1 ISPs having more POPs over a larger geographic area. I observed no significant correlation

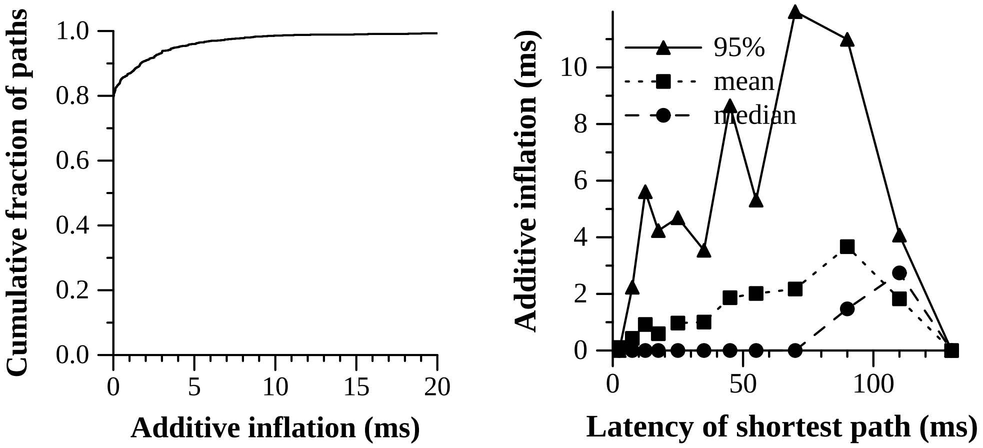


Figure 7.13: Path inflation due to intra-domain routing policy, when compared to shortest latency paths. The left graph shows the CDF of path inflation, and the right one shows path inflation as function of the latency of the shortest path.

between path inflation due to topology and the ISP's geographic presence, suggesting that ISPs use similar backbone network designs on various continents.

7.3.1.2 Inflation from intra-domain routing policy

With an understanding of the baseline of path inflation due to topology, I now turn to the path inflation caused by intra-domain routing policies. The same analysis is applied, though the path inflation is now the difference in length between the chosen path and the shortest-latency path in the topology. If large, this indicates a pattern of routing around congestion. If small, it suggests that the network is adequately provisioned to carry packets along the shortest-latency paths.

Figure 7.13 shows the results of the analysis. The overall median, mean, and 95th percentile were 0, 1, and 4 ms. The CDF on the left shows that 80% of the intra-domain paths are not inflated at all. This suggests that intra-domain traffic engineering is consistent with latency-sensitive routing; ISPs try to minimize latency while tweaking weights to balance traffic across backbone links. The graph on the right plots inflation as a function of the shortest path latency using the binning described above. (The length of the shortest path is used for binning.) Path inflation rises as the shortest path latency increases. This makes intuitive sense, as there are more acceptable paths between distant cities, allowing more traffic engineering. Closer investigation of paths beyond 50 ms revealed that most appear to be caused by attempts to balance load across two or more transcontinental links. The decrease near the right edge of the graph is an artifact of having very few city pairs in that bin, most of which are connected by a nearly direct path.

7.3.1.3 Summary

The absence of circuitous paths is an argument for the existence of adequate provisioning, but that conclusion is uncertain. If ISP operators were content to let packets drop or be queued significantly within their networks, they might not route around congestion, and no circuitous intra-domain paths would be observed. ISPs, however, typically guarantee a certain level of service to customers, limiting how many packets may be dropped or delayed. The use of direct paths does not tell by how much these links may be over-provisioned: it may be the

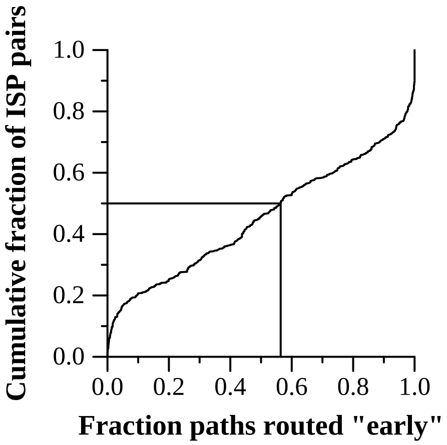


Figure 7.14: The prevalence of the early-exit routing policy. For ISP pairs where I observed more than one peering link, this graph shows the fraction of paths I observed that were routed to the earliest exit. The median is 57%, meaning that most ISP pairs choose the early exit most of the time.

case that some ISPs operate near capacity while others do not. Finally, the use of direct paths in steady state does not imply that routing is not used to alleviate temporary congestion. Despite these limitations, it seems more reasonable to assume ISP networks are well-provisioned, and the intra-domain routing policy analysis I present is consistent with that assumption.

7.3.2 Peering Routing Policies

Peering routing policies represent the relationships between pairs of ISPs as manifest in how they direct packets. They also represent a poorly-understood interface between inter-domain and intra-domain routing: peering policies are implemented in the inter-domain routing protocol, BGP, but affect router-level paths with many of the same constraints found in intra-domain routing. The confluence of long-term, pairwise business agreements with short-term traffic engineering goals makes peering routing policy a particularly complex domain.

Early-exit routing is the default, simplest to implement peering routing policy. Early-exit requires only a knowledge of which downstream ISP to choose to reach a destination, and it forwards packets along at what should be minimum cost for the upstream ISP. Some operators and researchers suggested ISPs also often used late-exit routing: where the upstream ISP carries traffic as close to the destination as possible before passing it to the downstream. To my knowledge, no one has previously quantified the prevalence of these routing policies. These policies are described in more detail in Section 2.2.2 and how I classify them appears in Section 6.4.

The goal of this study were to understand the prevalence of these two policies and to understand what patterns might emerge: whether only some ISPs used late-exit or whether it was only used some of the time. The complexity of the picture I found surprised me: peering policies are diverse and are not uniformly applied. I now describe the results of this analysis.

7.3.2.1 Prevalence of early-exit

I first evaluated, for every pair of ISPs, what fraction of observed paths were routed using the “early” exit. Each ingress point in an upstream ISP has an “early” exit to the downstream ISP. The fraction of observed paths that

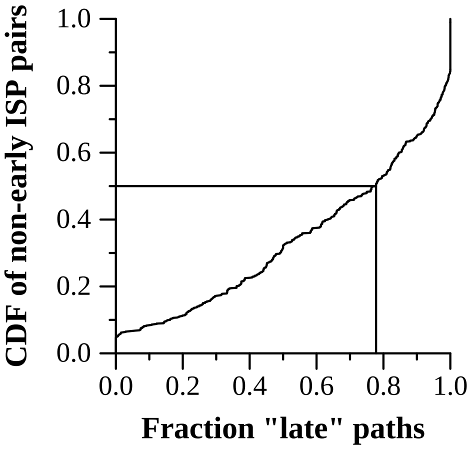


Figure 7.15: Of ISP pairs that choose paths that are not always consistent with “early-exit” routing, this graph shows the fraction of paths that are “helped” along toward the destination, in which the upstream ISP carries the traffic closer to the destination than necessary. Each point represents an ISP pair, the fraction of “helped” paths is the x-axis value. ISP pairs with a high fraction of “helped” paths are likely to use “late-exit” routing. When the fraction of “helped” paths is small, however, another policy, such as “load-balancing,” may be a more likely explanation.

were “early” is simply the number that reach the downstream through that “early” exit. ISP pairs that appear to have only a single peering point (where every path must be early) are ignored in this analysis.

Figure 7.14 presents the results. *While half of all ISP pairs route at least half of their prefixes using early-exit, many do not, hinting at widespread cooperation among ISPs.* Additionally, the amount of path engineering varies. To the right of the graph, 20–30% of ISP pairs appear to use only early-exit; to the left, 10–20% route with a policy other than early exit (likely late-exit, but that is not shown here) and in between is a wide range. This diversity is surprising—it suggests that early-exit may be an adequate model of network routing, but there is significant complexity.

7.3.2.2 Prevalence of late-exit

I next determine how many ISP pairs have the upstream network carry traffic closer to the destination. This definition of late-exit, presented in Section 6.4.1, is loose in that the exit chosen may not be the closest to the destination, it only need be closer.

Figure 7.15 shows a cumulative distribution of the fraction of late-exit paths. The cooperation among ISPs is apparent in this graph, too: *most ISP pairs that do not use early-exit route over 75% of paths closer to the destination.* For instance, Qwest routes almost all prefixes destined to Genuity using a late-exit that is independent of the source. A few ISP pairs route only a few paths close to the destination: suggesting either micro-engineering of a few prefixes or a limited load balancing policy. Again, peering routing policies are surprisingly diverse, but when early-exit is not used, late-exit routing appears to be the most likely alternative.

7.3.2.3 Peering policies of tier-1 ISPs

Table 7.1 summarizes the apparent peering routing policy of tier-1 ISPs. Each pair of ISPs is classified once for each direction, because peering routing policy need not be (and often is not) symmetric. I use the method

Table 7.1: Summary of observed peering relationships between tier-1 ISPs. Both early- and late-exit policies are common, and not simply the work of a few ISPs. Peering policies vary widely, even between neighbors of the same ISP. Empty cells are ISPs that peer in Route Views but were not observed in the traceroute dataset.

From To	ATT	ATT-Disc	Cable Wireless	Colt	DTAG	Eqip	Genuity	Genuity-europe	Global Crossing	Globix	Hong Kong Telecom	Level3	Netrail	Open Transit	Qwest	Sprint	Teleglobe	Telia	UUNet	Verio	Williams	XO
ATT	–	!!	!	–	!	x	●	–	●	●	–	!	○	!	!!	x	●	!	x	●	!	x
ATT-Disc	○	–	–	–	–	–		–			–	–	–	–	x	–	–	–	–			
Cable Wireless	!!	–	–	x	!	○	!	–	x	–	–	●	●	x	!!	●	!!	!	●	●	●	!
Colt	–	–		–	–	–		–	–				–		–	–	–	–	–	○	–	–
DTAG		–	○	●	–	–	○					!	–	○				○	–	○		
Eqip		–		–	–	–		–				–	–		○		–	–				
Genuity	x		●	○	●	●	–	!	x	x	!!	●	○	x	!!	●	●	x	●	●	●	●
Genuity-europe	–	–	–	–		–	●	–			○		–		–	○	–	–	–	–	–	–
Global Crossing	!!	!!	!!	○	x	○	!!	!	–	○	○	●	○	!	●	!!	!!	○	!	!!	○	x
Globix			–							–			–		–	○	–	–	x			
Hong Kong Telecom	–	–	○	○		●	●	○	●		–		–		○	○	–	–	–	○		
Level3	x	–	!	●	!!	○	!!	○	●	●	●	–	○	x	!	●	●	!	!!	!	x	●
Netrail	○	–	●	–	○	–	○	–	○	–	○	○	–	○	●	○	–	–		–	–	
Open Transit		–	○	○		○	○					!		–	○	○		●	○		–	
Qwest	x	!	x	○	○	○	!!	–	!	–	!	x	●	x	–	x	x	x	!!	x	x	x
Sprint	x	–	x	○	●	○	x		●	!	!	x	–	x	●	–	x	!!	!!	x	!	!!
Teleglobe	!!	–	●	–	○	–	●	–	●	–	–	!!	–	!!	x	!!	–	○	!!	x	–	
Telia	!!	–	!!	○	!!	x	!!	●	!!	–	○	!!	○	!!	!!	!!	!	–	!!	!!	!!	–
UUNet	●	–	x	–	–	!	●	–	●	!	○	●	○	●	!	x	x	●	–	●	!	●
Verio	●	○	x	!!	!!	x	●	○	!	x	!	!	–	x	●	●	x	●	!	–	x	x
Williams Comm			!!	–		○		–	○		○	○	–	–	●	●	–	○	!!	○	–	
XO	●		!	–			●	–	○		○	○		○	○	●		–	!	○	○	–

!!	“Late exit,” often	(15%)	●	Source-dependent “early-exit”	(19%)	x	Engineered, but not “late”	(13%)
!	“Late-exit,” sometimes	(10%)	○	Single peering seen	(42%)	–	Peering not present in RouteViews	

described in Section 6.4.1 to classify the peering routing policy between ISP pairs. Pairs labeled *Late-exit, often* deliver more paths late (closer to the destination) than early. Those labeled *Late-exit, sometimes* deliver at least 5% of paths late. *Source-dependent early-exit* refers to ISP pairs for which fewer than 5% of paths are routed specially. A threshold makes the classification robust to anomalous routes during failures. *Single peering seen* have a single peering link in one location; many of these connect ISPs on different continents. Policy has no apparent effect for these ISP pairs, though some of these ISP pairs may have undiscovered peering points for reasons presented in Figure 6.10. *Engineered, but not late* are ISP pairs with many paths that are not early, but at least 1/3 of the paths do not cross a peering closer to the destination. Finally, some ISP pairs peer in Route Views [80] but no peering link appeared in the dataset.

Some ISPs, such as Telia and Global Crossing, appear to use late exit more often than others. In all, this figure demonstrates the diversity of pairwise peering policies and that "early-exit" rarely sums up routing policy between tier-1 ISPs.

I found asymmetry in the peering routing policies used overall and associated with the tier of the ISP. Of the ISP pairs classified in both directions as early or late, the likelihood of a late exit policy is 17%, but the likelihood that an ISP reciprocates late exit is only 8%. The relationship between tier-1 and tier-2 ISPs is asymmetric. Half (82 of 178) of the tier-1 to tier-2 edges showed a late-exit policy compared to only 36% (32 of 88) of the tier-2 to tier-1 edges. I did not observe major differences in the policies used in different parts of the world.

7.4 Summary

In this chapter, I presented and analyzed the structured, policy-annotated ISP topologies I measured. These analyses took advantage of the unique properties of structured, router-level topologies and the new potential of routing policy inference. This analysis scratches the surface of what outsiders can learn about the design of networks.

I draw several conclusions from my analysis of topologies. Although the distribution of router out-degree is skewed and highly-variable over its small range, the out-degree distribution within these measured ISP topologies is not, technically, heavy-tailed. Most POPs are small but most routers are in POPs with more than 16 routers. Each of the ISPs I studied is likely to peer widely with a few other ISPs, and peer in only a few places with many other ISPs. The ISPs I studied connect to hundreds of other ASes in a few central cities.

This study of network routing policy in the Internet provided the following conclusions. Intra-domain routing is consistent with shortest-latency routing and consistent with adequate provisioning of the network. Peering routing policies are diverse, even to different peers of the same ISP. Peering routing policies are likely to be asymmetric: late-exit is unlikely to be reciprocated. Finally, peering routing policy may be applied to avoid congestion on peering links, a use of routing policy I did not observe in intra-domain routing.

Chapter 8

CONCLUSIONS AND FUTURE WORK

In this chapter, I review the thesis and contributions of this dissertation and describe future work in network measurement.

8.1 Thesis and Contributions

In this dissertation, I supported the following thesis: *outsiders can use routing and debugging information to efficiently measure accurate and structured, router-level ISP topologies and infer ISP routing policies*.

The goal of this work was to develop techniques to understand the structure and configuration of the Internet despite its decentralization and narrow interfaces. The following two key insights supported this work. First, the narrow interfaces in the network were designed not to prevent others from knowing about the operation of ISPs, but only to prevent ISPs from depending on the internal details of others. Second, information from various sources can be synthesized to help determine which measurements are likely to be redundant. By aggressively avoiding redundant measurements, outsiders can discover the router-level network topologies and routing policies used by ISPs: new information about network design that has proven useful for research.

Measured network topologies should foster the development of better protocols, systems, and tools, and allow them to be evaluated in a way that provides confidence in the result. New systems may be able to exploit common patterns of network design, such as hierarchy and local redundancy, and be evaluated in a way that considers the heterogeneity of network designs. This information is especially important because changes to the Internet are changes to a live system, which complicates experimentation, prototyping, and widespread deployment.

The contributions made by this dissertation included:

An approach to accurate and efficient ISP network topologies mapping. The Rocketfuel approach to network mapping is based on the selection of, out of all measurements that could be taken, those likely to provide new information. The techniques combine BGP information with prior measurements to choose relatively few measurements to be taken. This approach balances completeness and efficiency to produce an accurate map. Treating efficiency as a primary goal enabled the use of hundreds of public traceroute servers as vantage points: a measurement platform much larger than used previously.

The techniques of traceroute selection, alias resolution, and DNS name interpretation realized the philosophy of Internet measurement espoused in this dissertation: aggressively avoid unnecessary measurement using previously-measured information as a guide, and apply this efficiency to increase accuracy. These techniques measured maps approximately seven times more detailed than prior efforts, in the ISPs being studied, yet use fewer measurements.

A set of techniques for inferring intra-domain and peering routing policies. I showed how to infer intra-domain and peering routing policy, recovering a compact yet predictive representation of how paths are selected. The key insight that enables the inference of routing policy is that paths *not* taken through the topology expose routing policy decisions. I further showed that once intra-domain routing policy is known, it can be used to infer peering routing policy. This work is the first to measure and study intra-domain and peering routing policies systematically.

The measured ISP network topologies and routing policies and an analysis of their characteristics. I measured two large datasets, one of the router-level topologies of 10 ISPs collected simultaneously, and a second of the POP-level topologies of 65 ISPs. The structured, router-level topologies were collected with significantly more accuracy than previous efforts. The POP-level topologies provided input to a routing policy analysis that supported the first look at intra-domain and peering routing policies across several ISPs. Taken together, the policy-annotated, structured ISP network topologies measured in this work have provided the research community a new means to study various topics.

8.2 Future Work

The work described in this dissertation enables two directions of future research. The primary directions for network measurement are in the integration of diverse measured information to assist validation and further measurement, and in the inference of additional properties of network topologies and designs.

8.2.1 Coordinated Internet Measurement

The ideal goal of network measurement is the discovery of all properties of all links and routers at all times. A giant step toward this goal is enabled by advancements along two independent research efforts: those that seek to measure properties of links including capacity [38, 63, 72], loss [73], and queuing [6, 73], and those efforts that seek to measure network maps including those described in Section 3.1.2 and this dissertation. Because so many researchers are involved in network measurement, I believe that progress can best be made by coordinating effort: allowing researchers to extend, repeat, and improve the work of others.

I envision a “measurement blackboard” that supervises the execution of measurements, archives and aggregates their results, and exports raw data and summary views including network topologies. Unlike Route Views and Skitter, which have shown the value of continuously archived wide-area measurements, the measurement blackboard includes such diverse information as the performance of links, the location of nodes, and routing policies, as well as raw measurements in the form of packet traces and traceroute output. Unlike the Internet Traffic Archive [62] and the proposed SIMR system [3] for indexing and archiving diverse Internet measurements, new measurements will build on the fresh, intermediate results of others. Unlike large monitoring systems like RIPE-NCC [53] or the recently proposed M-coop [117] peer-to-peer monitoring system, the measurement blackboard will be persistent and allow measurements to build upon each other. To support such tightly coupled measurement, the measurement blackboard will execute measurement and inference programs that researchers write to a common API. Scriptroute [116] running on PlanetLab [97] provides a suitable large-scale measurement platform. This architecture of researchers contributing extensions to a narrow core is similar to how the network simulator *ns* [89] has facilitated improved experimentation and comparison.

The challenging components of a measurement blackboard are infrastructure to: lower the barrier to running large-scale network measurements from many vantage points, lower the barrier to running network measurements periodically and unattended, schedule measurements of individual links to avoid conflicts, and most importantly, plan efficient measurement of each link in the topology.

Annotating a map with measured properties becomes practical when the basic map of the topology helps decide how to take detailed measurements. This observation is an application of the philosophy of network measurement in this dissertation: avoid unnecessary measurements. The network is much smaller than all-to-all measurements. By referring to a map in the process of measurement, it becomes possible to measure the properties of links just once. Avoiding repeated, redundant measurements of the same links may make measuring the capacity of every link in the network practical.

The potential for improving the accuracy of network measurement tools is one component of the motivation for this effort. The accuracy of network measurement tools has been debated [67, 120] and many tools have a tradeoff between the number of packets needed and the accuracy of the result [38, 73]. Various tools have been evaluated on a few known paths [38, 94] but the continuous change in the network means that these experiments are not easily repeated. One approach to validation enabled by a unified framework for network measurement is to compare the results of various tools—this may be most useful when evaluating a more efficient approach to measuring some property. Another approach is to include "true" values made available by research networks as if they were measured—although these true values are likely to be incomplete, having diverse, distant network links to measure would help evaluate the accuracy of network measurement tools.

This integrated effort aims to supply the networking research community with more accurate and appropriate models of network operation [45]. That is, to allow researchers to study diverse topics with a well-annotated, reasonably accurate network topology.

A few practical lessons from this dissertation can help shape the design of a collaborative platform. First, a database worked well for coordination between independent measurement tasks. Recall that ISP mapping consisted of traceroute collection, alias resolution, and DNS name decoding. Each task was independent, and the coordination offered by transactions simplified implementation. Second, *time* should be treated explicitly. Without time, it is difficult to recognize change. For example, when measurements are inconsistent, as in Section 5.5, it is unclear whether the error is because information is incorrect or simply out of date. Third, integrating raw measurement with the inference of additional properties in the same framework is useful for supporting further extension, as I found with inferred locations and intra-domain routing policy.

8.2.2 Inferring New Properties

Although my work has shown that ownership, geographic location, router roles, and network routing policies can be measured and inferred by outsiders, several important properties of the topology remain to be measured. To measure these properties in the context of a network map requires addressing the same high-level challenges of opaque interfaces and scale as faced measuring topology. I believe techniques can be developed to measure such properties as client location, failure, utilization, and layer 2 topology, although much research remains to address practical issues and validate techniques. The following text sketches a possible approach for each of these properties.

8.2.2.1 Client location

Clients create a demand for traffic. Although network mapping projects focus on collecting an accurate picture of the core of the network, clients ultimately shape the network—more clients create more demand, inducing ISPs to add capacity. Andersen *et al.* [7] showed that prefixes that share BGP behavior mirror the underlying topology. BGP updates are already collected and stored at several sites. The traceroutes collected as part of network mapping could also be used to determine where clients attach in the topology.

Knowing how much address space lies in a location alone is insufficient for suggesting their demand on the network. For example, smaller prefixes appear to be more densely populated with clients and servers [77]. As a result, knowing where prefixes connect should be augmented with an estimate of the number of active hosts attached to the network—potentially from an unobtrusive, but possibly biased, passive analysis at Web servers.

8.2.2.2 Failure

Failures generally last only a few minutes [61], so externally observing failures, let alone determining where they occur, is daunting. However, as another instance of heterogeneity in the Internet, some links are more failure prone than others [61], which means that if the links and routers that are likely to fail can be identified, measurements can be focused on those.

Several primitives can detect node and link failures. A simple approach to detecting long term failure is to analyze changes in the measured network: removal and replacement of a link or router suggests a temporary failure. To measure short-term failures, however, requires new approaches. First, when packets traverse a path that is inconsistent with routing policy, a failure may be the cause. Inferring the routing policy during the failure may show a link (or set of links around a failed node) as having “infinite” cost—exposing the failed component. This extends the approach of Dahlin *et al.* [35]. To detect node failure explicitly, probe traffic could monitor routers to observe the IP identifier described in Section 4.2. Because the IP identifier is usually implemented as a counter, this counter may be cleared when a router is rebooted or powered off, providing an externally-visible sign of failure.

The approach of inferring link failure through policy inference is distinct from the tomography-based failure inference methodologies of Nguyen and Thiran [86], Bejerano and Rastogi [13], and Horton and López-Ortiz [58]. These approaches combine direct probing of both ends of a network link with an analysis to find the minimum number of vantage points[1] that cover every link in a network topology. The use of direct probing to detect link failure assumes that paths are symmetric, which may be valid when vantage points are within an ISP and reverse path symmetry applies as in Section 6.1.4. Paths are frequently asymmetric, however, when crossing ISPs. Because my goal is to learn properties of networks as an outsider, I cannot rely on vantage points within the ISP.

8.2.2.3 Utilization and workload

The utilization of a network link is the fraction of the capacity of a link that is used. Tools like clink [38] can measure link capacity and tools like pathload [65] can measure the available (idle) capacity of a path, but researchers have yet to develop tools to measure the available capacity of individual links [102]. Measuring both the available capacity of a link and its total capacity can provide an estimate of link utilization.

The utilization of a link may be difficult to measure because it is a dynamic property with troublesome statistical properties [96]. Two approaches may estimate utilization. First, pathload may be extended using tailgating. Pathload uses an adaptive search to detect when it barely fills the idle capacity of a path. The available capacity of links before the bottleneck may be measured by modifying pathload’s link-filling traffic to expire at various hops in the network while small tailgating packets continue on to a receiver.[2] Links downstream of the bottleneck may be mapped by combining the results of other vantage points. Second, tools such as cing [6] and tulip [73] measure delay variation and loss to a router. It may be possible to analyze the distribution of these samples to estimate utilization. Alouf *et al.* [4] present an approach that models a link as an M/D/1/K queue, and uses queue length (which can be sampled by delay variation), loss rate, and capacity to solve for the capacity of the queue and link utilization. Results using these techniques can be validated against known workloads from cooperating ISPs.

8.2.2.4 Below IP

Internet mapping efforts that use traceroute are limited in their ability to discover the real, physical topology underlying the IP-level topology: MPLS, multi-access LANs, and other switched networks can appear as a mesh of virtual point-to-point links. Identifying these networks would allow a deeper understanding of physical connectivity: how many interfaces routers have and which IP-level connections between routers are likely to share properties such as congestion or failure.

The immediate goal in understanding layer two topologies is to simplify measured network maps that include switched networks. Common address allocation policies may provide a solution. Many point-to-point links in the core of the Internet (not dial-up PPP) are assigned addresses from /30 prefixes—address blocks with one address for either end of the link, a network address, and a broadcast address.[3] As a result, in a network full

[1] Vantage points are termed *beacons* in the formulation of the link fault monitoring problem.

[2] Hu and Steenkiste try the opposite approach in which small packets expire at each hop and large packets continue [59].

[3] Recently, operators have started to use /31 prefixes with the expectation that broadcast and network addresses are unnecessary on point-to-point links [108].

of point to point IP links, one half of the addresses will be observed as unavailable (those that end in 0 and 3 modulo 4). Observing an interface address that is not part of a point to point link indicates that a larger address block (such as a /29 or /28) is being used by a shared or switched network. This method would not detect a "switch" in the middle of a point-to-point link, but the result should simplify measured maps.

Also of interest is the underlying network switch topology. Prasad *et al.* [101] showed that store-and-forward switches have an observable effect on pathchar-like tools. The extra store and forward stage represented by an "invisible" switch doubles the extra serialization delay observed when increasing the size of single packet probes. This makes the link appear only half as fast as it is. If any tool measures greater available bandwidth than link capacity, a switch is indicated. Similarly, one might use traffic designed to contend for only certain segments of a hypothesized switched link, as proposed by Coates *et al.* [34]. Similarly, differences between geographic latency and measured link latency can imply detour routing at the link layer.

8.2.3 Summary

My work has scratched the surface of understanding network design and operation, and much work remains. The structured, router-level ISP topologies measured by my techniques provide a foundation for annotating additional properties, leading toward a comprehensive understanding of the Internet. My measurement of reasonably accurate ISP maps was enabled by the synthesis of different sources of information and a philosophy of taking only the measurements likely to contribute new information. These and similar maps can be used to better understand network operation, to build network protocols that exploit common patterns of network design, and to determine how best to improve the design of networks in the Internet.

BIBLIOGRAPHY

[1] Aditya Akella, Bruce Maggs, Srinivasan Seshan, Anees Shaikh, and Ramesh Sitaraman. A measurement-based analysis of multihoming. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 353–364, Karlsruhe, Germany, August 2003.

[2] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. In *Nature*, volume 406, pages 378–382, July 2000.

[3] Mark Allman, Ethan Blanton, and Wesley M. Eddy. A scalable system for sharing Internet measurement. In *Proceedings of Passive & Active Measurement (PAM)*, pages 189–191, Fort Collins, CO, March 2002.

[4] Sara Alouf, Philippe Nain, and Don Towsley. Inferring network characteristics via moment-based estimators. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1045–1054, Anchorage, AK, April 2001.

[5] Lisa Amini, Anees Shaikh, and Henning Schulzrinne. Issues with inferring Internet topological attributes. In *Proceedings of the SPIE ITCOM Workshop on Internet Performance and Control of Network Systems*, volume 4865, pages 80–90, Boston, MA, July 2002.

[6] Kostas G. Anagnostakis, Michael B. Greenwald, and Raphael S. Ryger. cing: Measuring network-internal delays using only existing infrastructure. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 2112–2121, San Francisco, CA, April 2003.

[7] David G. Andersen, Nick Feamster, Steve Bauer, and Hari Balakrishnan. Topology inference from BGP routing dynamics. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 243–248, Marseille, France, November 2002.

[8] Ran Atkinson and Sally Floyd, editors. IAB concerns and recommendations regarding Internet research and evolution. Internet Engineering Task Force Request for Comments RFC-3869, August 2004.

[9] Fred Baker. Requirements for IP version 4 routers. Internet Engineering Task Force Request for Comments RFC-1812, June 1995.

[10] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[11] Thorold Barker. Tiscali expands German holdings with AddCom. *Financial Times*, page 28, December 21, 2000.

[12] Tony Bates and Philip Smith. http://www.cidr-report.org/, August 2004.

[13] Yival Bejerano and Rajeev Rastogi. Robust monitoring of link delays and faults in IP networks. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 134–144, San Francisco, CA, April 2003.

[14] John Bellardo and Stefan Savage. Measuring packet reordering. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 97–105, Marseille, France, November 2002.

[15] Steven M. Bellovin. A technique for counting NATted hosts. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 267–272, Marseille, France, November 2002.

[16] Michel Berkelaar. lp_solve: Mixed integer linear program solver. ftp://ftp.ics.ele.tue.nl/pub/lp_solve/.

[17] Supratik Bhattacharyya, Christophe Diot, Jorjeta Jetcheva, and Nina Taft. Pop-level and access-link-level traffic dynamics in a Tier-1 POP. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 39–54, San Francisco, CA, November 2001.

[18] Ian Bikerton. KPNQwest's Ebone closes after failed rescue. *Financial Times*, page 28, July 3, 2002.

[19] Vincent J. Bono. 7007 explanation and apology. http://www.merit.edu/mail.archives/nanog/1997-04/msg00444.html, April 1997.

[20] Carsten Bormann, Carsten Burmeister, Mikael Degermark, Hideaki Fukushima, Hans Hannu, Lars-Erik Jonsson, Rolf Hakenberg, Tmima Koren, Khiem Le, Zhigang Liu, Anton Martensson, Akiiro Miyazaki, Krister Svanbro, Thomas Wiebke, Takeshi Yoshimura, and Haihong Zheng. Robust header compression. Internet Engineering Task Force Request for Comments RFC-3095, July 2001.

[21] Alan Borning, Bjorn Freeman-Benson, and Molly Wilson. Constraint hierarchies. *Lisp and Symbolic Computation*, 5(3):223–270, 1992.

[22] Andre Broido and kc claffy. Internet topology: connectivity of IP graphs. In *Proceedings of the SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, volume 4526, pages 172–187, Denver, CO, August 2001.

[23] Tian Bu and Don Towsley. On distinguishing between Internet power law topology generators. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 638–647, New York, NY, June 2002.

[24] Hal Burch and Bill Cheswick. Burch and Cheswick map of the Internet. http://research.lumeta.com/ches/map/gallery/isp-ss.gif, June 1999.

[25] Hal Burch and Bill Cheswick. Mapping the Internet. *IEEE Computer*, 32(4):97–98, 102, April 1999.

[26] Didier Burton and Philippe L. Toint. On an instance of the inverse shortest paths problem. *Mathematical Programming*, 53(1):45–61, 1992.

[27] Cable & Wireless completes acquisition of Exodus. Business Wire, February 1, 2002.

[28] Hyunseok Chang, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. On inferring AS-level connectivity from BGP routing tables. Technical Report UM-CSE-TR-454-02, University of Michigan, 2002.

[29] Qian Chen, Hyunseok Chang, Ramesh Govindan, Sugih Jamin, Scott J. Shenker, and Walter Willinger. The origin of power laws in Internet topologies revisited. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 608–617, New York, NY, June 2002.

[30] Bill Cheswick, Hal Burch, and Steve Branigan. Mapping and visualizing the Internet. In *Proceedings of the USENIX Annual Technical Conference*, pages 1–12, San Diego, CA, June 2000.

[31] Cisco Systems. *Internetworking Technologies Handbook*. Cisco Press, 4th edition, 2000.

[32] Cisco Systems. Using the traceroute command on operating systems. http://www.cisco.com/warp/public/105/traceroute.shtml, May 2004. Cisco Document 22826.

[33] kc claffy, Tracie E. Monk, and Daniel McRobb. Internet tomography. *Nature, Web Matters*, January 1999. http://www.nature.com/nature/webmatters/tomog/tomog.html.

[34] Mark Coates, Rui Castro, Robert Nowak, Manik Gadhiok, Ryan King, and Yolanda Tsang. Maximum likelihood network topology identification from edge-based unicast measurements. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 11–20, Marina del Rey, CA, June 2002.

[35] Michael Dahlin, Bharat Baddepudi V. Chandra, Lei Gao, and Amol Nayate. End-to-end WAN service availability. *IEEE/ACM Transactions on Networking*, 11(2):300–313, 2003.

[36] Christopher Davis, Paul Vixie, Tim Goodwin, and Ian Dickinson. A means for expressing location information in the domain name system. Internet Engineering Task Force Request for Comments RFC-1876, January 1996.

[37] Jay L. Devore. *Probability and Statistics for Engineering and the Sciences*. Duxbury Press, 4th edition, 1995.

[38] Allen B. Downey. Using pathchar to estimate Internet link characteristics. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 241–250, Cambridge, MA, September 1999.

[39] Paul Erdös and Alfréd Rényi. On random graphs. *Publicationes Mathematicae*, 6(26):290–297, 1959.

[40] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the Internet topology. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 251–262, Cambridge, MA, September 1999.

[41] Marwan Fayed, Paul Krapivsky, John W. Byers, Mark Crovella, David Finkel, and Sid Redner. On the emergence of highly variable distributions in the autonomous system topology. *ACM Computer Communication Review*, 33(2):41–49, April 2003.

[42] Nick Feamster, David G. Andersen, Hari Balakrishnan, and M. Frans Kaashoek. Measuring the effects of Internet path faults on reactive routing. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 126–137, San Diego, CA, June 2003.

[43] Nick Feamster and Jennifer Rexford. Network-wide BGP route prediction for traffic engineering. In *Proceedings of the SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, volume 4868, pages 55–68, Boston, MA, August 2002.

[44] Sándor P. Fekete, Winfried Hochstättler, Stephan Kromberg, and Christoph Moll. The complexity of an inverse shortest path problem. *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, 49:113–127, 1999.

[45] Sally Floyd and Eddie Kohler. Internet research needs better models. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, pages 29–34, Princeton, NJ, October 2002.

[46] Bernard Fortz and Mikkel Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 519–528, Tel Aviv, Israel, March 2000.

[47] Paul Francis, Sugih Jamin, Cheng Jin, Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. IDMaps: A global Internet host distance estimation service. *IEEE/ACM Transactions on Networking*, 9(5):525–540, October 2001.

[48] Vince Fuller, Tony Li, Jessica (Jie Yun) Yu, and Kannan Varadhan. Classless inter-domain routing (CIDR): an address assignment and aggregation strategy. Internet Engineering Task Force Request for Comments RFC-1519, September 1993.

[49] Lixin Gao. On inferring autonomous system relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9(6):733–745, December 2001.

[50] Lixin Gao and Feng Wang. The extent of AS path inflation by routing policies. In *IEEE Global Telecommunications Conference (GLOBECOM) Global Internet Symposium*, volume 3, pages 2180–2184, Taipei, Taiwan, November 2002.

[51] J. J. Garcia-Luna-Aceves and Shree Murthy. A path-finding algorithm for loop-free routing. *IEEE/ACM Transactions on Networking*, 5(1):148–160, 1997.

[52] Ehud Gavron. NANOG traceroute. ftp://ftp.login.com/pub/software/traceroute/beta/.

[53] Fotis Georgatos, Florian Gruber, Daniel Karrenberg, Mark Santcroos, Ana Susanj, Henk Uijterwaal, and René Wilhelm. Providing active measurements as a regular service for ISP's. In *Proceedings of Passive & Active Measurement (PAM)*, March 2001.

[54] Geoffrey Goodell, William Aiello, Timothy Griffin, John Ioannidis, Patrick McDaniel, and Aviel Rubin. Working around BGP: An incremental approach to improving security and accuracy in interdomain routing. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, pages 74–84, San Diego, CA, February 2003.

[55] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for Internet map discovery. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1371–1380, Tel Aviv, Israel, March 2000.

[56] Charles Hedrick. Routing information protocol. Internet Engineering Task Force Request for Comments RFC-1058, June 1988.

[57] Robert V. Hogg and Johannes Ledolter. *Applied Statistics for Engineers and Physical Scientists*. Macmillan, New York, 2nd edition, 1992.

[58] Joseph D. Horton and Alejandro López-Ortiz. On the number of distributed measurement points for network tomography. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 204–209, Miami, FL, October 2003.

[59] Ningning Hu and Peter Steenkiste. RPT: A low overhead single-end probing tool for detecting network congestion positions. Technical Report CMU-CS-03-218, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, December 2003.

[60] Young Hyun, Andre Broido, and kc claffy. Traceroute and BGP AS path incongruities. http://www.caida.org/outreach/papers/2003/ASP/, 2003.

[61] Gianluca Iannaccone, Chen-nee Chuah, Richard Mortier, Supratik Bhattacharyya, and Christophe Diot. Analysis of link failures in an IP backbone. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 35–47, Marseille, France, November 2002.

[62] The Internet Traffic Archive. http://ita.ee.lbl.gov/.

[63] Van Jacobson. Pathchar. ftp://ftp.ee.lbl.gov/pathchar/.

[64] Van Jacobson. Traceroute. ftp://ftp.ee.lbl.gov/traceroute.tar.Z.

[65] Manish Jain and Constantinos Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 295–308, Pittsburgh, PA, August 2002.

[66] Raj Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. John Wiley & Sons, 1991.

[67] Guojun Jin and Brian L. Tierney. System capability effects on algorithms for network bandwidth measurement. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 27–38, Miami, FL, October 2003.

[68] Thomas Kernen. traceroute.org. http://www.traceroute.org/.

[69] Anukool Lakhina, John Byers, Mark Crovella, and Peng Xie. Sampling biases in IP topology measurements. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 332–341, San Francisco, CA, April 2003.

[70] Anukool Lakina, John W. Byers, Mark Crovella, and Ibrahim Matta. On the geographic location of Internet resources. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 249–250, Marseille, France, November 2002.

[71] Mathew J. Luckie, Anthony J. McGregor, and Hans-Werner Braun. Towards improving packet probing techniques. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 145–150, San Francisco, CA, November 2001.

[72] Bruce Mah. Estimating bandwidth and other network properties. In *Internet Statistics and Metrics Analysis Workshop on Routing and Topology Data Sets: Correlation and Visualization*, San Diego, CA, December 2000.

[73] Ratul Mahajan, Neil Spring, David Wetherall, and Thomas Anderson. User-level Internet path diagnosis. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, pages 106–119, Bolton Landing, NY, October 2003.

[74] Ratul Mahajan, David Wetherall, and Thomas Anderson. Understanding BGP misconfiguration. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 3–16, Pittsburgh, PA, August 2002.

[75] Z. Morley Mao, Jennifer Rexford, Jia Wang, and Randy Katz. Towards an accurate AS-level traceroute tool. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 365–378, Karlsruhe, Germany, August 2003.

[76] Nils McCarthy. fft. http://www.mainnerve.com/fft/.

[77] Sean McCreary and kc claffy. IPv4 address space utilization. http://www.caida.org/outreach/resources/learn/ipv4space/, August 1998.

[78] Alberto Medina, Ibrahim Matta, and John Byers. BRITE: A flexible generator of Internet toplogies. Technical Report BU-CS-TR-2000-005, Boston University, 2000.

[79] Alberto Medina, Ibrahim Matta, and John Byers. On the origins of power-laws in Internet toplogies. *ACM Computer Communication Review*, 30(2):18–28, April 2000.

[80] David Meyer. University of Oregon Route Views project. http://www.routeviews.org/.

[81] David Moore, Colleen Shannon, Geoffrey M. Voelker, and Stefan Savage. Internet quarantine: Requirements for containing self-propagating code. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1901–1910, San Francisco, CA, April 2003.

[82] John Moy. OSPF version 2. Internet Engineering Task Force Request for Comments RFC-2328, April 1998.

[83] Katta G. Murty. *Linear Programing*. John Wiley & Sons, 1983.

[84] Akihiro Nakao and Larry Peterson. BGP feed configuration memo. Technical Report PDN–03–011, PlanetLab Consortium, April 2003.

[85] T. S. Eugene Ng and Hui Zhang. Predicting Internet network distance with coordinates-based approaches. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 1, pages 170–179, New York, NY, June 2002.

[86] Hung X. Nguyen and Patrick Thiran. Active measurement for multiple link failures diagnosis in IP networks. In *Proceedings of Passive & Active Measurement (PAM)*, pages 185–194, Antibes Juan-les-Pins, France, April 2004.

[87] H. Penny Nii. Blackboard systems, part one: The blackboard model of problem solving and the evolution of blackboard architectures. *AI Magazine*, 7(2):38–53, 1986.

[88] H. Penny Nii. Blackboard systems, part two: Blackboard application systems. *AI Magazine*, 7(3):82–106, 1986.

[89] UCB/LBNL/VINT network simulator - ns, 2000.

[90] David R. Oran. OSI IS-IS intra-domain routing protocol. Internet Engineering Task Force Request for Comments RFC-1142, February 1990.

[91] Venkata N. Padmanabhan and Lakshminarayanan Subramanian. An investigation of geographic mapping techniques for Internet hosts. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 173–185, San Diego, CA, August 2001.

[92] Jean-Jacques Pansiot and Dominique Grad. On routes and multicast trees in the Internet. *ACM Computer Communication Review*, 28(1):41–50, January 1998.

[93] Kihong Park and Heejo Lee. On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 15–26, San Diego, CA, August 2001.

[94] Attila Pásztor and Darryl Veitch. Active probing using packet quartets. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 293–305, Marseille, France, November 2002.

[95] Vern Paxson. End-to-end routing behavior in the Internet. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 25–38, Palo Alto, CA, August 1996.

[96] Vern Paxson and Sally Floyd. Wide-area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[97] Larry Peterson, Thomas Anderson, David Culler, and Timothy Roscoe. A blueprint for introducing disruptive technology into the Internet. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, pages 59–64, Princeton, NJ, October 2002.

[98] Graham Phillips, Scott Shenker, and Hongsuda Tangmunarunkit. Scaling of multicast trees: Comments on the Chuang-Sirbu scaling law. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 41–51, Cambridge, MA, September 1999.

[99] James Politi. Savvis trumps Gores to snap up Cable and Wireless' US assets. *Financial Times*, page 34, January 23, 2004.

[100] Jon Postel, editor. Internet protocol specification. Internet Engineering Task Force Request for Comments RFC-791, September 1981.

[101] Ravi S. Prasad, Constantinos Dovrolis, and Bruce A. Mah. The effect of layer-2 switches on pathchar-like tools. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, pages 321–322, Marseille, France, November 2002.

[102] Ravi S. Prasad, Margaret Murray, Constantinos Dovrolis, and kc claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *IEEE Network*, 17(6):27–35, November 2004.

[103] Bruno Quoitin, Steve Uhlig, Cristel Pelsser, Louis Swinnen, and Olivier Bonaventure. Interdomain traffic engineering with BGP. *IEEE Communications Magazine*, 41(5):122–128, May 2003.

[104] Pavlov Radoslavov, Hongsuda Tangmunarunkit, Haobo Yu, Ramesh Govindan, Scott Shenker, and Deborah Estrin. On characterizing network topologies and analyzing their impact on protocol design. Technical Report CS-00-731, USC, 2000.

[105] Rajeev Rastogi, Yuri Beribart, Minos Garofalakis, and Amit Kumar. Optimal configuration of OSPF aggregates. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 874–882, New York, NY, June 2002.

[106] Yakov Rekhter and Tony Li. A border gateway protocol 4 (BGP-4). Internet Engineering Task Force Request for Comments RFC-1771, March 1995.

[107] Yakov Rekhter, Robert G. Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear. Address allocation for private internets. Internet Engineering Task Force Request for Comments RFC-1918, February 1996.

[108] Alvaro Retana, Russ White, Vince Fuller, and Danny McPherson. Using 31-bit prefixes on IPv4 point-to-point links. Internet Engineering Task Force Request for Comments RFC-3021, December 2000.

[109] Ronald L. Rivest. The MD5 message-digest algorithm. Internet Engineering Task Force Request for Comments RFC-1321, April 1992.

[110] Eric C. Rosen, Arun Viswanathan, and Ross Callon. Multiprotocol label switching architecture. Internet Engineering Task Force Request for Comments RFC-3031, January 2001.

[111] Stefan Savage, David Wetherall, Anna Karlin, and Thomas Anderson. Practical network support for IP traceback. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 295–306, Stockholm, Sweden, August 2000.

[112] Philip Smith. BGP multihoming techniques. North American Network Operator's Group (NANOG), June 2003. Tutorial slides: http://www.nanog.org/mtg-0306/smith.html.

[113] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer. Hash-based IP traceback. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 3–14, San Diego, CA, August 2001.

[114] Neil Spring, Mira Dotcheva, Maya Rodrig, and David Wetherall. How to resolve IP aliases. Technical Report 04–05–04, University of Washington Department of Computer Science and Engineering, May 2004.

[115] Neil Spring, David Wetherall, and Thomas Anderson. Reverse-engineering the Internet. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, pages 3–8, Cambridge, MA, November 2003.

[116] Neil Spring, David Wetherall, and Thomas Anderson. Scriptroute: A public Internet measurement facility. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS)*, pages 225–238, Seattle, WA, March 2003.

[117] Sridhar Srinivasan and Ellen Zegura. Network measurement as a cooperative enterprise. In *International Workshop on Peer-to-Peer Systems (IPTPS)*, pages 166–177, Cambridge, MA, March 2002.

[118] W. Richard Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison Wesley, 1994.

[119] John Stewart, Tony Bates, Ravi Chandra, and Enke Chen. Using a dedicated AS for sites homed to a single provider. Internet Engineering Task Force Request for Comments RFC-2270, January 1998.

[120] Jacob Strauss, Dina Katabi, and Frans Kaashoek. A measurement study of available bandwidth estimation tools. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 39–44, Miami, FL, October 2003.

[121] Lakshminarayanan Subramanian, Sharad Agarwal, Jennifer Rexford, and Randy H. Katz. Characterizing the Internet hierarchy from multiple vantage points. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 618–627, New York, NY, June 2002.

[122] Laksminarayanan Subramanian, Venkata N. Padmanabhan, and Randy H. Katz. Geographic properties of Internet routing. In *Proceedings of the USENIX Annual Technical Conference*, pages 243–259, Monterey, CA, June 2002.

[123] Nina Taft. The basics of BGP routing and its performance in today's Internet. RHDM (Resaux Haut Debit et Multimedia), High-Speed Networks and Multimedia Workshop, May 2001. Tutorial slides: http://ipmon.sprint.com/pubs_trs/tutorials/Taft_BGP.pdf.

[124] Nina Taft, Supratik Bhattacharyya, Jorjeta Jetcheva, and Christophe Diot. Understanding traffic dynamics at a backbone POP. In *Proceedings of the SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, volume 4526, pages 150–156, Denver, CO, August 2001.

[125] Hongsuda Tangmunarunkit, Ramesh Govindan, and Scott Shenker. Internet path inflation due to policy routing. In *Proceedings of the SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, volume 4526, pages 188–195, Denver, CO, August 2001.

[126] Hongsuda Tangmunarunkit, Ramesh Govindan, Scott Shenker, and Deborah Estrin. The impact of routing policy on Internet paths. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 736–742, Anchorage, AK, April 2001.

[127] Renata Teixeira and Jennifer Rexford. A measurement framework for pin-pointing routing changes. In *Proceedings of the ACM SIGCOMM Network Troubleshooting Workshop*, pages 313–318, Portland, OR, August 2004.

[128] Michael C. Toren. tcptraceroute. http://michael.toren.net/code/tcptraceroute/.

[129] Paul Traina, Danny McPherson, and John G. Scudder. Autonomous system confederations for BGP. Internet Engineering Task Force Request for Comments RFC-3065, February 2001.

[130] Ronald E. Walpole and Raymond H. Myers. *Probability and Statistics for Engineers and Scientists*. Prentice Hall, Englewood Cliffs, New Jersey, 5th edition, 1993.

[131] Feng Wang and Lixin Gao. Inferring and characterizing Internet routing policies. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 15–26, Miami, FL, October 2003.

[132] Bernard M. Waxman. Routing of multipoint connections. *IEEE Journal of Selected Areas in Communications*, 6(9):1617–1622, December 1988.

[133] Xiaowei Yang. NIRA: A new Internet routing architecture. In *Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA)*, pages 301–312, Karlsruhe, Germany, August 2003.

[134] Bin Yao, Ramesh Viswanathan, Fangzhe Chang, and Daniel Waddington. Topology inference in the presence of anonymous routers. In *Proceedings of the IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 353–363, San Francisco, CA, April 2003.

[135] Tao Ye and Shivkumar Kalyanaraman. A recursive random search algorithm for large-scale network parameter configuration. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 196–205, San Diego, CA, June 2003.

[136] Artur Ziviani, Serge Fdida, José F. de Rezende, and Otto Carlos M. B. Duarte. Toward a measurement-based geographic location service. In *Proceedings of Passive & Active Measurement (PAM)*, pages 43–52, Antibes Juan-les-Pins, France, April 2004.

VITA

Neil Spring was born in San Diego, California. He received his Bachelor of Science degree in computer engineering from the University of California, San Diego in 1997, and his Master of Science degree in computer science from the University of Washington in 2000. His research interests include network measurement, adaptive scheduling of distributed applications, and the security of open systems.