Endpoint-Based Routing Strategies for Improving Internet Performance and Resilience

Aditya Akella

CMU-CS-05-183

September 29, 2005

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Srinivasan Seshan, Chair
Bruce M. Maggs
Hui Zhang
Scott Shenker, UC Berkeley

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

This research was sponsored by the National Science Foundation under grant nos. CNS-0435382, ANI-0092678, and ANI-0085920, the US Army Research Office under contract no. DAAD19-02-1-0389, and the US Air Force Research Laboratory under grant no. F306029910518. It was also funded through a generous grant from the IBM Corporation. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Internet routing, Border Gateway Protocol, performance bottlenecks, overlay networks, route control, congestion, Moore's law, Internet topology

Abstract

Internet access speeds of large enterprises and educational institutions have improved dramatically over the past few years. However, this higher-speed connectivity is still ineffective at providing end-users with good download performance and robustness from service interruptions. This arises due to the prevalence of constrained links with little spare capacity inside Internet Service Provider (ISP) networks.

In this dissertation, we first investigate the location, latency and traffic load characteristics of network links that limit the Internet performance of well-connected end-networks. More importantly, we show how end-networks can employ a clever Internet route selection technique, called Multihoming Route Control, to avoid these performance bottlenecks and obtain much better Internet performance. Using Internet-scale measurements conducted over Akamai's content distribution infrastructure, we show that by multihoming to three ISPs, and intelligently scheduling transfers across the ISPs, an end-network could potentially improve its Internet round-trip times (RTTs), throughputs and reliability by up as much as 30%.

We also compare the Internet performance and reliability from route control against more powerful route selection paradigms such as overlay routing. We show that the RTTs and transfer speeds from multihoming are within 5-10% of overlay routing. While multihoming cannot offer the nearly perfect resilience of overlays, we show that it can eliminate almost all failures experienced by a singly-homed end-network. We also describe the design and performance evaluation of a route control system that can be deployed by large multihomed enterprises. We show that, in practice, simple route control techniques can offer Web performance within 10% of the optimal performance from multihoming.

Finally, we investigate whether, in the future Internet, techniques such as route control or overlay routing can still provide good end-to-end performance in the face of higher access speeds and a vastly different traffic mix. We show that the structure of the Internet (i.e., a power law degree structure at the ISP level), together with the routing protocol (i.e., BGP), will convert certain keys portions of the network into persistent bottlenecks. We then consider modifications to the ISP-level interconnections to guarantee good end-to-end performance in the future Internet.

We believe that the contributions in this thesis significantly advance the state-of-the-art of techniques for improving Internet performance and resilience. Further, this dissertation highlights important guidelines for the design of inter-domain routing protocols and peering architectures.

To Nanna, Amma, Chandu and Shuchi

Acknowledgments

I consider myself immensely fortunate to have pursued my Ph.D. at Carnegie Mellon University. Carnegie Mellon provided me with access to some of the best minds in Computer Science and great resources for research. Carnegie Mellon takes exceptionally good care of its students, so much so, I felt it was "home away from home". The five years I spent at CMU have been fun-filled and memorable.

I was quite fortunate to have Srinivasan Seshan as my Ph.D. adviser. Srini gave me the necessary freedom to work on topics of my choice and, despite his busy schedule, never shied away from spending time with me when I needed his input and attention. My Ph.D. research was also shaped by constant interactions with Bruce Maggs, Hui Zhang and Anees Shaikh. Bruce was instrumental in helping me gain access to Akamai's data and in formulating key sub-problems in this thesis. Hui helped me immensely broaden the scope of my research by bringing in the 100x100 network perspective. Anees offered great insight when interpreting our measurement results, and in fleshing out a design for the route control system presented in this dissertation. Jeff Pang and Arvind Kannan were of great help in conducting some of the key measurement experiments and simulations, and analyzing the results. In addition, my past interactions with Scott Shenker, Richard Karp, Ion Stoica, Christos Papadimitriou, Peter Steenkiste, Ramesh Sitaraman and Balachander Krishnamurty have significantly influenced my approach to research in general.

Finally, I would like to thank my close friends, Luis von Ahn, Laura Dabbish, Mahesh Bandi and "Bunny" for spicing up my graduate years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past decade, Internet access speeds of large organizations, such as enterprises, data centers and universities have improved tremendously. Several years ago, organizations and end-users alike were limited to a few Kbps of network connection speeds. Since then, technological breakthroughs such as Fiber Optic links, coupled with growing demand for bandwidth from newer applications, have fueled rapid improvements in endpoint access speeds. Now-a-days, most organizations deploy very high speed connections, often 100Mbps or more, to ensure robust performance of their "mission critical" network applications.

End-user access speeds have also improved substantially over the same time-period. In Figure 1.1, we show the evolution of home access speeds in the United States. Market studies predict that the adoption of 100Mbps broadband home connections in the U.S. will begin as early as the year 2010! In fact, home users in Japan and South Korea already enjoy impressive connection speeds in the order of several tens of Mbps [22, 116].

When access speeds are limited, the Internet experience of end-users is tightly coupled with their Internet connection speeds: A 28Kbps modem line can be expected to provide inferior download speeds and availability than a DSL link (especially when the client opens multiple Web connections in parallel). Similarly, Web pages take longer to load and data transfers take longer to complete with a 1.5Mbps home DSL connection, than a T3 connection (45Mbps) in an office environment. Naturally, this leads one to think that improved access speeds in the near future will drastically improve the Internet experience of end-users.

But, is this really true? Lets suppose that an enterprise upgraded its high-speed Internet connection from 10Mbps to 100Mbps, or even 1Gbps. Informally, we refer to end-networks with high-speed Internet access as being *well-connected*. Then, the above question translates to whether the Internet access performance of end-users in well-connected organizations, specifically their observed download speeds and resilience from service interruptions, would improve proportionately.

Of course, higher speed connections will offer better-than-reasonable performance and resilience. But it is unclear if future well-connected end-users and end-networks will see the *bang for their buck*, i.e., matching performance for their higher access speeds. Here, and henceforth, when the

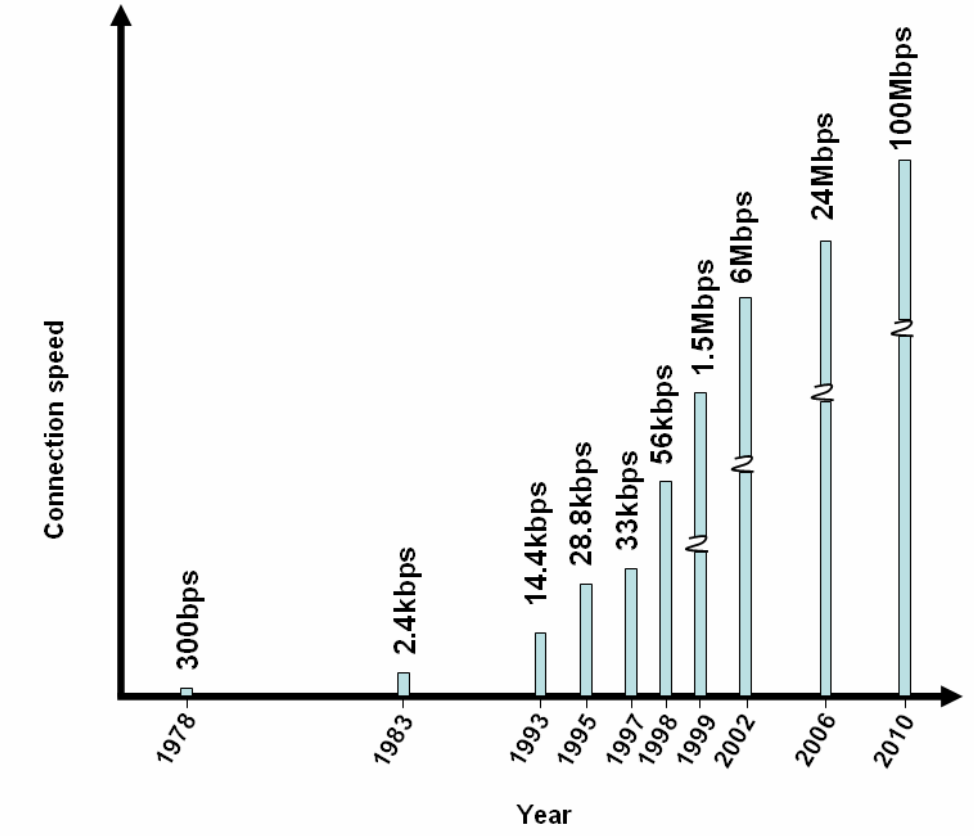


Figure 1.1: Evolution of home access speeds: We show when various home Internet access technologies (specifically, the corresponding access speeds) were adopted in the U.S. We also show the expected adoption dates for higher speed home access, such as 24Mbps and 100Mbps.

context is unambiguous, we use the term "performance" to collectively refer to Internet download speeds, response times for Internet transfers and resilience from service interruptions, as observed by network endpoints.

Unfortunately, as the connection speed of a network endpoint grows, its performance will most likely *not* improve proportionately. In fact, it is highly likely that a 1Gbps uplink, for example, will offer no better download speeds and response times to an endpoint than a 100Mbps link. As we discuss next, this arises due to the prevalence of key performance "road-blocks", or *bottlenecks* which limit the Internet experience of well-connected endpoints.

Our focus in the forthcoming discussion, and in the rest of the dissertation, will be on end-networks, such as enterprises, universities and Web server hosting centers: we will use the term "endpoints" to collectively refer to these network entities. The observations and proposals we make for end-networks in this dissertation can also be extended to individual end-hosts.

1.1 Road-blocks to Efficient Internet Performance

There are several key bottlenecks that prevent a well-connected endpoint from realizing the best possible performance from its high-speed access connections. In general, these bottlenecks can be broadly classified into two categories: access and non-access bottlenecks.

The first category includes performance bottlenecks that are under the control of the network endpoints themselves, a simple example being ill-configured software stacks at end-hosts. Internet transfer speeds, response times, and even resilience, to some extent, may intrinsically depend on the software stacks employed by end-hosts. For example, several past studies have identified the impact of the receive buffer size setting in endpoint Transmission Control Protocol (TCP) stacks on the throughput achieved by TCP connections [101, 84]. Small receive buffer sizes constrain the amount of data a transmitter can simultaneously relay to a receiver. As another illustration, older implementations of TCP, such as TCP Reno, are less resilient to packet drops in the network than latter deployments such as TCP SACK. As a result, clients employing TCP Reno may observe poorer network performance than SACK clients, especially when the traffic load on the network is high.

An important feature of access bottlenecks is that, being under the administrative control of endpoints, they can be easily overcome by means of simple local upgrades, such as updating system software on host machines, and careful configuration. With sufficiently high speed access links, up-to-date software, and perfect configuration, then, the performance of well-connected, well-managed endpoints is limited only by *non-access bottlenecks*.

1.1.1 Non-access Bottlenecks

Non-access bottlenecks will persist irrespective of the access speeds or software upgrades at network endpoints. Essentially, these bottlenecks arise from inefficient design, operation and management of the wide-area Internet as a whole, or of key portions of it. To overcome these bottlenecks, therefore, requires coordinated network-wide upgradation and management—a much more formidable task than local improvements. A key example of such non-access bottlenecks are constrained wide-area hot-spot links. Other examples include failures in critical network-wide services, such as the Internet's Domain Name System (DNS). More often than not, factors such as DNS failures tend to preclude or delay the start of new Internet transfers. We do not discuss such factors here. Instead we focus on bottlenecks that may impact both new as well as ongoing Internet transfers.

Internet transfers will have to share network resources, such as raw link capacities and router buffers, with millions of other flows in the network. If the volume of this competing data traffic is low, the only major constraint on the performance of a data transfer is likely to be the raw capacity of the network links it traverses (assuming no software limitations). Often, since the raw speeds of most access links are much lower than those of links elsewhere in the network, i.e., of "non-access" links, the performance of the data transfer is limited primarily by the source or the destination connection speeds.

However, many non-access links today carry several hundreds of megabits of data traffic per second, especially during peak hours. Internet backbone providers, which own these non-access links, employ a variety of mechanisms to protect the links from traffic overload. But the unpredictability of traffic as well as conflicting traffic management goals of competing providers often leave keys links in the network very congested, and with little "head room" or spare capacity for additional traffic. A simple fix to alleviating congestion at such hotspots would be to add additional capacity wherever necessary. However, this approach is not economically viable. Moreover, selective local improvements to link speeds may simply push hotspots to other network locations.

In other words, the performance of transfers initiated by well-connected endpoints depends critically on the volume of competing traffic on each non-access link they traverse, and the variation in the traffic volumes with time. Non-access links running at close-to-full capacity may drive endpoint transfer speeds close to zero, force packet drops, and cause connections to time-out or even fail. In general, we will refer to non-access links with heavy traffic load that limit the Internet performance of well-connected endpoints as *wide-area bottleneck links*. When the context is clear, we will also use the more generic terms non-access bottlenecks or wide-area bottlenecks to refer wide-area bottleneck links.

In order to overcome wide-area bottlenecks and obtain satisfactory Internet performance, endpoints may require intrinsic support from the network. For example, upon traffic overload at a certain link, the network may identify the link as a hotspot and divert endpoint traffic away from the link, in a manner completely transparent to the endpoint itself. Given that the Internet as whole is composed of thousands of links, and, at any time, many of these links may have enough spare capacity to carry diverted traffic, such a seamless rerouting of endpoint traffic is indeed plausible. So, in practice, does the Internet divert endpoint traffic away from hotspot network locations, as and when they arise? The answer to this question is, unfortunately, *no*.

Ideally, if the Internet were to support seamless diversion of traffic from hotspots then this should be handled, at least in part, by the Internet's *routing protocol*. That the Internet is does not do so arises from fundamental limitations of the routing protocol. Next, we briefly overview the Internet's routing protocol suite and examine why it cannot help endpoints circumvent wide-area bottlenecks.

1.1.2 Internet's Routing Protocol: BGP

Data traffic between any two points in the network will have to traverse several routers and links. The Internet's routing protocol suite determines the exact sequence of routers and links to be traversed. A quick overview of the Internet routing protocol follows.

Internet routing overview. The Internet's routing infrastructure is composed of several thousands of routers and links. The ownership of this infrastructure, and its operational and management responsibilities, are distributed across several independent administrative domains, called Internet

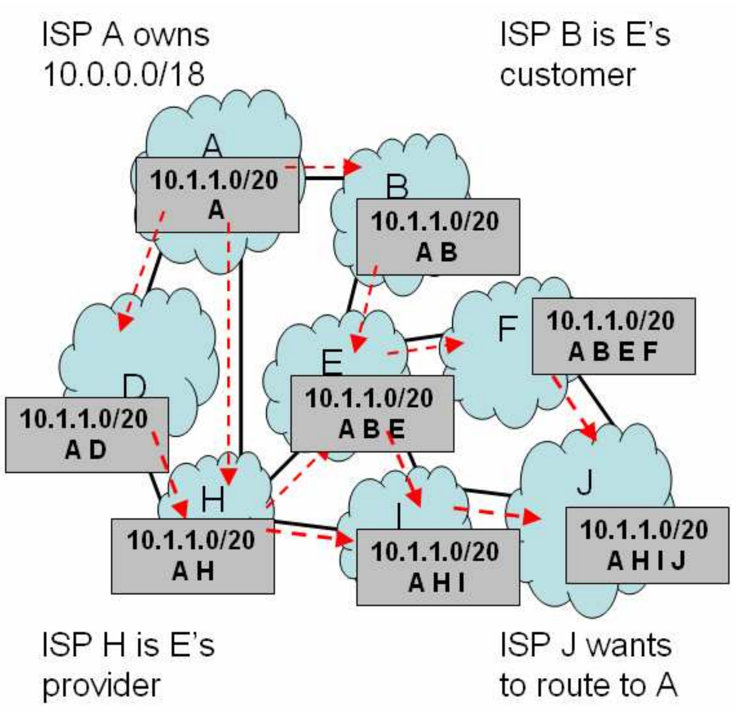


Figure 1.2: BGP operation: An example showing the propagation of BGP reachability and routing information across ISPs. The arrows show the propagation of routing announcements.

Service Providers (ISPs) or Autonomous Systems (ASes). Inside a domain, each ISP runs an "intra-domain" routing protocol across its routers and links to determine how traffic should be routed across the domain. Typically, when computing these routes, ISPs try to optimize the usage of their routing infrastructure. For example, an ISP may route traffic in such a manner that no particular link in its network carries an unduly large amount of traffic. This practice is commonly referred to as *traffic engineering*.

When data traffic traverses multiple domains, ISPs employ the Border Gateway Protocol (BGP) to exchange traffic. Figure 1.2 illustrates the operation of BGP. In this example, ISP J wishes to communicate with ISP A. A owns the Internet address block 10.0.0.0/18. In BGP, A announces reachability to the address block it owns to each of its neighbors D, H, E and B. Each neighbor, in turn announces this information further downstream, appending its domain number (called the autonomous system number, or ASN) to the announcement. Consider ISP H. As the announcements propagate through the network, H receives two announcements for 10.0.0.0/18: one through ISP D, and another directly through ISP A. BGP path selection mechanisms stipulates that H prefer the route with fewer ISPs (i.e., the announcement from A), over the other route available via ISP D. H announces this route further downstream. Now, consider ISP E. Both the announcements it receives for 10.0.0.0/18, through ISPs H and B, have the same number of intermediate ISPs. In this case, E prefers the more cost-effective route. For example, due to commercial arrangements with its neighbors, routing via H may require E to pay H (in which case H is referred to as the provider of E). In contrast, B may be the customer of E, and therefore, routing via B will boost E's revenue. As a result, E prefers to use the route E→B→A to reach 10.0.0.0/18. Announcements propagate across ISPs in this manner, eventually reaching ISP J. After the final selection of the route has been made, data from J to A will traverse the path J→I→H→A. As mentioned above, within each domain, ISPs employ traffic engineering mechanisms to manage their network. BGP-based routing in the Internet is also commonly referred to as *policy routing* due to the explicit support in the protocol for commercial arrangements between ISPs.

BGP characteristics. Notice that the ISP-level route selected by BGP is completely agnostic to the existence of non-access bottlenecks, and the performance or availability of the ISP-level path itself. The primary goal in BGP is to provide course-grained reachability information, while honoring the economic considerations of intermediate ISPs. While BGP does favor policy-compliant paths with fewer ASes in them, this choice is very static and does not reflect the *quality* of the path in any manner. Also, the requirement for paths to be policy-compliant could arbitrarily "inflate" them, resulting in long, circuitous routes.

Furthermore, in BGP, each ISP only exposes what it believes to be the single "best" route for a destination to each of its neighbors. Therefore, an endpoint with a single ISP connection has a single path per destination via its ISP. In short, the information propagated across ISPs in BGP is heavily filtered and summarized. This property, in fact, helps BGP to scale to thousands of networks. However, it gives rise to serious performance inefficiencies, discussed next.

Problems with BGP. The above characteristics of BGP imply that if a performance or availability problem arises in one of the ISPs along the path to a destination, the end-network will either have to "live with it", or wait until BGP selects an alternate path to the destination. Past research studies have shown that the latter process may, in fact, take a very long time. For example, Labovitz et al. [62] study re-convergence in BGP and show that BGP may take up to several minutes to recompute and propagate new routes after a failure. This implies that BGP lacks the necessary flexibility to route endpoint traffic around network hotspots and failures in a timely and effective manner.

Several past studies have quantified the impact of this "rigidity" in BGP-based routing on the Internet performance and availability experienced by endpoints. Feamster et al. [39] show that endpoints relying on BGP-based routing may experience connectivity outages lasting up to several minutes, or sometimes even hours. Savage et al. [99] quantify the sub-optimality in endpoint transfers speeds and Web response times arising from relying on BGP-based routes. The authors show that congestion and heavy traffic load on BGP paths can be avoided by dynamically selecting alternate, non-policy-compliant Internet paths. This could result in substantially improved transfer speeds and response times compared to traditional Internet routing. Tangmunarunkit et al. [110] and Spring et al. [104] highlight another crucial inefficiency in Internet routing: adhering to traffic exchange policies among ISPs often forces the paths between pairs of Internet endpoints to be close to 1.5 times as long as the direct, "as-the-bird-flies" paths.

In short, these past studies establish that BGP routing significantly limits the ability of well-connected end-networks to obtain good Internet performance and resilience from interruptions. This leads one to ask if well-connected network endpoints can employ special mechanisms that work in conjunction with Internet routing to extract the optimal performance from their high-speed Internet connections.

1.1.3 The Problem and Our Approach

In this dissertation, we first complement the above studies of inefficiencies in Internet performance and availability with an investigation of traffic load on wide-area links. We study the extent to which wide-area bottleneck links limit the performance of data transfers involving well-connected endpoints. We find that a significant fraction of non-access links carry high volumes of traffic, leaving very little spare capacity (under 10Mbps on many occasions). This, in turn, places an upper bound on the speeds that transfers initiated by well-connected endpoints may attain. Since BGP is ineffective at enabling a quick recovery from these performance problems, in this thesus, we investigate the usefulness of mechanisms that work in conjunction with BGP to overcome the bottlenecks.

Specifically, we seek to answer the following central question:

> *What mechanisms can well-connected end-networks employ to circumvent performance and availability problems in the non-access portion of the Internet, and improve their Internet experience?*

As mentioned above, this issue has been partly addressed by past research studies. However, most of these approaches require support from either ISPs, in terms of cutting-edge routing policies or traffic engineering mechanisms, or from special purpose Internet-wide infrastructures, to explicitly bypass BGP-determined routes and divert end-network traffic from heavily-loaded network hot-spots. In contrast, our focus is on mechanisms that require neither special support from network-wide infrastructure, nor any modifications to BGP or to common routing policies. Rather, we study mechanisms that can simply be deployed by endpoints at their local networks (e.g., at enterprise and university access routers, or inside very high-speed home DSL gateways) and be seamlessly integrated into the existing wide-area Internet infrastructure.

The specific mechanism we study is called "Multihoming Route Control" and involves the end-networking buying Internet connections from multiple ISPs serving the city it is located in. More importantly, we allow the end-network the ability to intelligently schedule its transfers across the ISP connections so as to avoid performance glitches in ISPs, whenever they arise. The end-network does not control how its ISPs route its traffic; rather, it can only control which ISP carries traffic to a particular destination at any time. The ISPs themselves may use BGP to route packets toward their destinations. In this sense, multihoming route control does not require any modification to the current BGP protocol or to network infrastructure.

We conduct extensive active and passive measurements across an 68-node Internet-wide testbed to quantify the performance benefits to end-networks from multihoming route control. We also use these measurements to compare multihoming route control against past proposals requiring special-purpose infrastructures, and show that our simple approach, in fact, offers similar performance as these more powerful approaches. We also present an implementation of multihoming route control for deployment in multihomed enterprise settings. We show that simple principles, such as regularly monitoring ISP performance and minimally relying on the historical performance of ISPs, are very effective in helping endpoints realize the potential benefits of multihoming in practice.

Also, we investigate whether mechanisms such as multihoming route control will be effective in the future Internet at all, in the face of growing end-user access speeds, higher traffic loads and shifting usage patterns. We study evolutionary patterns of the Internet's ISP level interconnection and show that the Internet's structure, together with BGP-style routing, render it fundamentally incapable of supporting good endpoint performance in the future. In light of this observation, we propose simple rules for altering interconnections between ISPs to guarantee robust future endpoint performance.

1.2 An Integrated Approach to Optimizing Internet Performance

Optimizing and improving Internet performance has been the holy grail of the networking research community for well over two decades. There have been numerous proposals for fine-tuning Internet behavior, to bring Internet performance and reliability closer to that of other engineered systems such as cars, and the telephone network. However, researchers in the past have approached the Internet performance optimization problem in a piece-meal fashion, attacking specific facets it, and coming up with interesting solutions to key sub-problems. Examples of such past approaches abound: studies of routing pathologies, intra and inter-domain traffic engineering, proposals for modifying and improving routing protocol behavior, specifically BGP, and overlay routing.

In contrast with these past approaches, this dissertation takes a more integrated approach to the problem: we first identify a primary cause for poor endpoint performance; then, we propose and evaluate a simple mechanism that gives endpoints the necessary control to overcome performance glitches; further, we compare our proposal against past well-established approaches to optimizing network performance and show that our proposal is both extremely simple and as effective as more complex approaches; finally, we also look into the future and ask if any of the existing proposals, be it multihoming or any other scheme, will be effective when the network grows in size and usage.

We believe that taking a "systemic" view of the problem, and analyzing various different aspects of it, lends our observations and mechanisms longer "shelf life". We feel that, owing to our approach, the key lessons from this thesis—i.e., encouraging richer first hop connectivity; enabling endpoint control over routing for performance; and leaving the rest of the network simple and untouched—will continue to apply to future Internet architectures, as well as to other problem domains such as wireless home networking.

1.3 Dissertation Outline

The rest of this dissertation is organized as follows. In Chapter 2, we go over past approaches for optimizing end-to-end performance in the Internet. Specifically, we discuss three classes of mechanisms—ISP-based, third-party infrastructure-based and end point-based—and put the contributions of this dissertation in perspective of these earlier approaches. In Chapter 3, we present a large-scale measurement study of performance bottlenecks in the wide-area network. We present a classification of these bottlenecks in terms of their location and latency, and discuss their impact on end-to-end performance. Building on these observations, in Chapter 4, we show how end-networks can employ multihoming route control to overcome the bottlenecks and improve their Internet performance. We also investigate how many, and which, ISPs endpoints must employ to realize optimal performance improvements. Further, in Chapter 5, we contrast the performance improvements from route control against past third-party infrastructure-based approaches. Specifically, we ask whether the limited routing flexibility of multihoming can provide comparable benefits as the arbitrary flexibility of the past approaches. Next, in Chapter 6, we ask whether the potential benefits from route control can indeed be realized in practical deployment scenarios. We implement and evaluate several route control mechanisms and outline important "best common" route control practices. In Chapter 7, we ask whether techniques such as route control will continue to provide good performance in the future network. Specifically, we investigate the impact of certain growth and operational trends of the Internet on its ability to support good performance in the face of growing end-user access speeds and new user applications. Finally, in Chapter 8, we present our conclusions and discuss issues for further study.

Chapter 2

Background and Approach

Optimizing and improving Internet performance and availability has been a hot topic of research for several years. Several proposals have been made for improving endpoint network stacks, such as TCP buffer-size tuning [57], better congestion control algorithms [11] and improved loss recovery mechanisms [71, 25]. Approaches for improving application-level performance, specifically, the performance of Internet Web servers are also well-studied (for example, better strategies for accepting incoming connections [26], improving Web response time using caching [54] and client characterization-based Web server adaption [61]).

Our focus in this dissertation is on non-access performance bottlenecks and approaches to optimizing Internet performance in face of such bottlenecks. Such approaches have been considered in both the research and commercial worlds, and some are even employed widely today. In general, these proposals can be classified into three broad categories:

1. **ISP-based:** These are mechanisms adopted by ISPs to cater to traffic traversing their networks. Examples include *intra-domain* and *inter-domain traffic engineering* mechanisms. A special class of mechanisms in this category includes approaches for *performance-aware wide-area Internet routing*. These special mechanisms often require global coordination across Internet ISPs.

2. **Internet-wide infrastructure-based:** These approaches involve the deployment of an application-level infrastructure across several ISPs. Such an infrastructure is typically operated and managed by a commercial third-party service provider. Examples include *overlay network*-based mechanisms. Multiple competing overlay service providers may deploy individual overlay infrastructures. The main purpose of an overlay infrastructure is to explicitly divert subscriber traffic from congested wide-area links or unavailable routes.

3. **End point-based:** These mechanisms are the simplest of the three. They require the subscriber network to buy Internet connectivity from 2-3 ISPs serving its city. These approaches also require deployment of special devices at the subscriber networks. However, these mechanisms do not require any further support from the ISPs themselves, or from special Internet-

	Example	Optimizes end-to-end performance?	Who deploys/ manages?	Fine-grained optimization?	BGP-compliant?	Deployment ease?
ISP-based	Inter-domain traffic engineering	No	Single ISP or neighboring ISPs	No	Yes	Need ISP co-operation
Internet-wide infrastructure-based	Overlay routing	Yes	Third-party provider	Yes	No	Need third-party deployment
Endpoint-based	Multihoming route control	Yes	Endpoint	Yes	Yes	Endpoint deploys unilaterally

Table 2.1: Approaches to optimizing Internet performance: This table shows the distinguishing features of common approaches to circumventing network-level performance bottlenecks.

wide infrastructures. Moreover, they do not require any modification to the Internet's wide-area protocol (i.e., BGP). Examples include *route control appliances*.

The distinguishing features of the above approaches are summarized in Table 2.1. We discuss these approaches in further detail next.

2.1 ISP Traffic Engineering

Approaches for optimizing the performance of ISP networks, commonly referred to as *intra-domain traffic engineering*, have been around for a very long time. For example, the ARPANET of the 1970s and 1980s employed adaptive routing across the network. In this setting, network routers in the network made routing choices based on the current state of the network. ARPANET routers maintained information about current (estimated) delays to other routers and forwarded packets along the path with the minimum estimated delay [72]. Further modifications were made to the routing protocol, for example, by incorporating a new routing metric which was based on a joint function of both hop counts and network delays, so as to minimize traffic fluctuations due to variations in network delays [58]. The key drawback of these "load-sensitive" approaches to traffic engineering was their reliance on the availability of fresh information about network state to compute network routes: stale network state could cause all network routers to avoid a common set of congested links leading to oscillations and congestion on other network links.

Modern day approaches to traffic engineering within ISP networks treat the problem as one of "network management": Network routers simply compute routes based on static link weights. The ISP network management system, in turn, sets the parameters based on a network-wide view of traffic and topology. To be precise, the static link weights are first set to be directly proportional to the distance between routers and inversely proportional to the capacity of the link between the routers. The static weights are fine-tuned by the network management system based on estimated network conditions, to directly minimize metrics such as the maximum link utilization.

To track network traffic, an ISP must first keep estimates of the traffic traversing its network.

In general, a *traffic matrix* describes the volume of data transmitted between a pair of routers in an ISP's network. Despite the centrality of traffic matrices to traffic engineering, today's routers provide little support for measuring them accurately. As such, the state-of-the-art in techniques for traffic matrix estimation is essentially to derive the underlying traffic matrix from measurements of link loads (obtained from Simple Network Management protocol (SNMP)) and ISP routing configurations. However, since there are far fewer routers and links in an ISP network than router-pairs, the estimates of the traffic matrix obtained in the above manner are highly error-prone. The high variability of traffic across ISP networks also contributes to the inaccuracy in traffic matrix estimates. A comprehensive survey of traffic matrix estimation techniques and related traffic engineering mechanisms may be found in [47, 96, 52].

Upon estimating the traffic matrix, an ISP employs the estimates to set links weights and route traffic across its network so as to optimize the usage of its network links and routers. The routing weights may also be set to satisfy performance requirements of select subscribers. ISPs typically configure link weights on the networks on a coarse time-scale: For example, a large backbone ISP may schedule daily maintenance of its network between midnight and 4 AM, when the new links weights may be computed and re-optimized to improve network congestion. Details of intra-domain traffic engineering mechanisms, proposals and extensions may be found in [18, 19, 16, 120].

In *inter-domain traffic engineering*, similar techniques as above are employed by two neighboring ISPs to manage the load on the links they share (i.e., peering links). This has only recently received the attention of the research and network operator community (see [90, 40] for complete details). The key challenge in inter-domain traffic engineering is to ensure effective inter-domain traffic management without revealing internal network information to neighboring networks. ISPs often hold their internal routing mechanisms and topology sacrosanct, and a poorly configured traffic engineering mechanism might reveal crucial hints regarding the internal policies of an ISP to its neighbor. Also, while intra-domain traffic engineering primarily dealt with selecting paths based on link metrics, in inter-domain traffic engineering an ISP will have to consider commercial arrangements with all its neighboring ISPs, along with link metrics, to determine the best paths. Moreover, effective inter-domain traffic engineering is made all the more challenging by the fact that the Internet's wide-area routing protocol, BGP, does not convey explicit information about the quality of paths. Therefore, most existing approaches to inter-domain traffic engineering explore minimal modifications to BGP to enable support for inter-domain traffic engineering. Due to these constraints, most ISPs are yet to adopt the latest proposals for inter-domain traffic engineering.

More importantly, both the afore-mentioned ISP-based mechanisms—intra- and inter-domain traffic engineering—are ineffective at improving the end-to-end Internet performance of endpoints due to two additional reasons. First, these approaches rely on traffic matrix estimation which, as mentioned earlier, is error-prone, and usually conducted on a daily time-scale. Therefore, these techniques may not be responsive enough, from the perspective of endpoints, to overcome unexpected performance or availability problems.

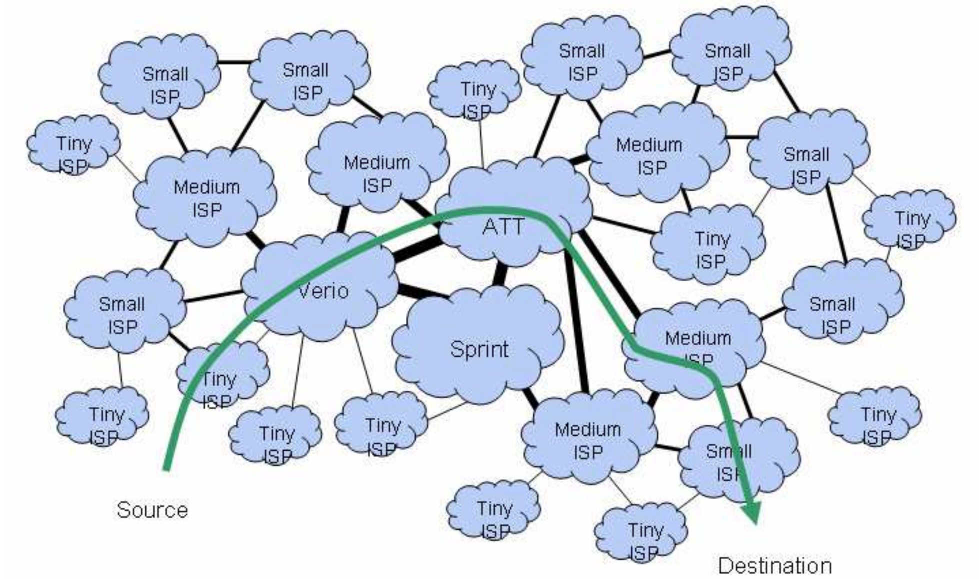


Figure 2.1: An end-to-end Internet flow: A typical Internet flow may traverse several ISPs end-to-end.

Second, these approaches are targeted at traffic confined to a single ISP or pairs of neighboring ISPs. In practice, traffic in the Internet could traverse multiple ISPs and peering locations, end-to-end. An example of a typical Internet flow is shown in Figure 2.1. The flow traverses multiple ISPs along its path, and distinct pairs of neighboring ISPs may have very different commercial arrangements, making coordinated traffic exchange between ISPs very challenging. The "local" performance optimizations performed by individual ISPs, therefore, may not improve the "global" end-to-end performance of Internet traffic. Recent research studies have proposed modifications to BGP that make cooperative traffic exchange across multiple ISPs easier than it is today. For example, the NIRA system [117] advocates incorporating support for AS-level source-routing into BGP. This gives both end-networks and ISPs greater control over the exact sequence of ISPs traversed by their data traffic. In turn, this feature makes it easier to configure new, inter-domain traffic engineering-friendly routing policies. However, since approaches such as NIRA require major changes to BGP, it may be several years before they are widely adopted.

2.2 Overlay Routing

As mentioned in Chapter 1, Internet routing in itself cannot support good end-to-end performance and resilience. Also, ISP-based traffic engineering mechanisms are not effective either, as mentioned in the previous section. To address these inefficiencies, and to guarantee good end-to-end Internet performance, several researchers have advocated deploying special purpose Internet-wide infrastructures to provide support for explicitly diverting subscriber traffic from highly congested

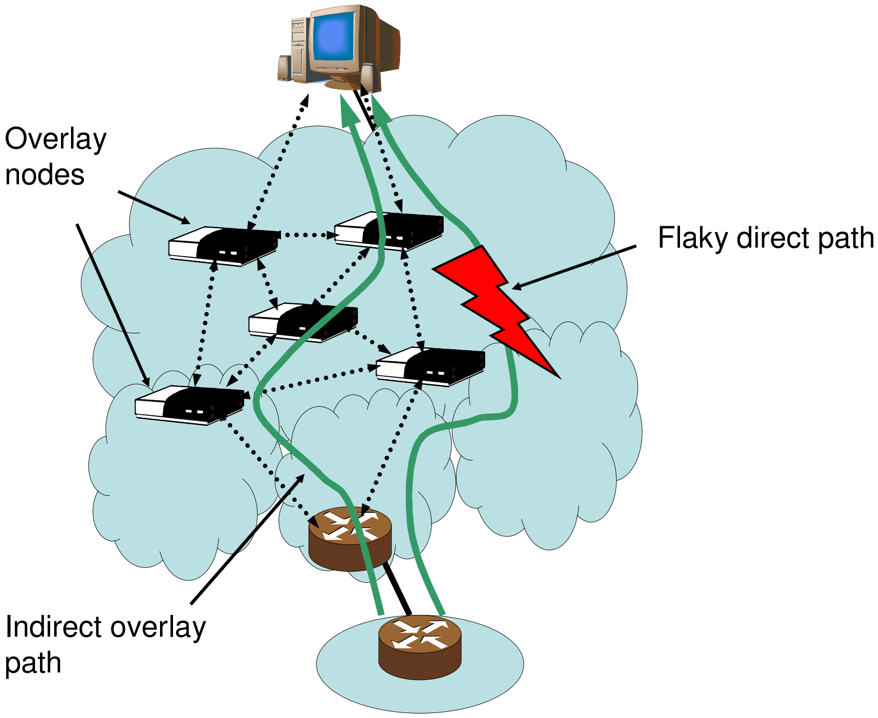


Figure 2.2: Overlay routing: This figure illustrates an overlay routing scenario.

network locations. The special-purpose infrastructures are called *Overlays* and these techniques are broadly referred to as *Overlay routing mechanisms*.

In overlay routing, a third-party overlay service provider deploys a large collection of overlay nodes in major cities across several ISPs throughout the Internet. These overlay nodes regularly monitor the performance and availability of paths between themselves and to various Internet destinations (using active probes, such as `ICMP Pings`, or passive measurements of existing data transfers between the nodes). The instantaneous performance and availability information is, in turn, accessible to the subscribers of the overlay service. The subscriber can than employ this information to obtain better Internet performance, as follows: Whenever the subscriber's default Internet route to a destination (as determined by BGP) does not offer the expected performance or availability, the subscriber can immediately route its traffic via an alternate path using the overlay nodes. This is shown in Figure 2.2. The subscriber, in effect, hands its traffic over to the overlay network, which routes the traffic across select overlay nodes, in a hop-by-hop manner, toward the destination. Notice that the route traversed by the subscriber traffic between neighboring overlay nodes in an overlay path must still conform to BGP policies. Therefore, the overlay nodes, by forming an application-level network among themselves, "stitch" multiple such BGP routes on-the-fly to yield better performing alternative paths to their subscribers. Notice also that overlays provide end-networks a great deal of flexibility in selecting quite arbitrary paths through the Internet.

Two famous research studies advocating overlay routing are the Detour system (see [99]) and the Resilient Overlay Network (RON) testbed (see [14]). In the Detour study, Savage et. al [99] study the impact of the inefficiencies of wide-area routing on end-to-end performance in terms of round-trip time (RTT), loss rate, and throughput. Using observations drawn from active measurements between public traceroute server nodes, they compare the performance on default Internet (BGP) paths with the potential performance from using alternate overlay paths. This work shows that for a large fraction of BGP paths measured, there are alternate overlay paths offering much better throughputs, RTTs, and loss rates. Andersen et. al advocate deploying RONs to address problems with BGP's fault recovery times, which have been shown to be on the order of tens of minutes in some cases [14]. RON nodes regularly monitor the availability of paths between each other, and use this information to dynamically select direct or indirect end-to-end paths. RON mechanisms are shown to significantly improve the availability of end-to-end paths between the overlay nodes, and to bring failure recovery times to within a few seconds.

In addition, a few commercial instantiations of overlays for improving web download performance have been proposed recently, such as Akamai SureRoute [4]. At regular intervals (called "sampling intervals"), the Akamai SureRoute system [4] replicates client data transfers across multiple overlay paths. The systems then selects the overlay path offering the fastest performance to the client, and employs the path for all subsequent transfers involving the client until the next sampling interval. SureRoute is offered as a resilience-enhancing service to both Web content providers, as well as large enterprise customers.

A natural concern regarding overlays is that, with wide-spread deployment and usage, the vast flexibility provided by overlays may actually result in worse network-wide congestion and oscillations in traffic. Indeed, past theoretical studies have highlighted examples of scenarios where allowing end-networks arbitrary flexibility in routing could substantially worsen the average performance of the network [97, 59]. For Internet-like scenarios, however, researchers have shown that wide-spread deployment of overlay routing does not necessarily result in worse overall performance. In fact, the average end-user performance with overlay routing is significantly better that that achieved with traditional routing protocols [89]. To summarize, there are few hurdles to the wide-spread deployment of overlay networks. However, very few end-networks today subscribe to overlays for optimizing their Internet performance [24]. Further, most of these end-networks employ overlays for a small but important fraction of their transfers, such as mission critical financial transactions.

We now discuss another key drawback of overlay-based approaches. A common feature of overlay-based approaches is that although the path between neighboring overlay nodes is BGP-compliant, the end-to-end path across the overlay network could arbitrarily violate commercial policies between ISPs. Imagine, for example, an overlay path consisting of the following sequence of ISPs: $A \rightarrow B \rightarrow C$ (with overlay nodes placed in ISPs A, B and C). Assume further that ISP B is a customer of ISPs A and C (e.g., A and C may be much larger ISPs, with wider global reach, and B may be a smaller regional ISP; Connecting to both A and C enables B to reach a wide range of Internet destinations). BGP's routing policies dictate that a customer network shall not provide transit for traffic from one provider destined to another provider network (this is clearly not in the best interest of the customer, since he would be paying both his providers for provider transit to their traffic!). It is easy to see that the above overlay path directly violates this policy. In such cases of violation, the overlay service provider compensates for ISP B's loss of revenue via explicit payment. In turn, this increases subscription and usage costs for an overlay client, especially if the client network uses the overlay for regular, high-volume Web access.

More importantly, however, it is unclear if the high degree of flexibility provided by overlay networks to overcome BGP-driven routes and policies is, in fact, *necessary* to help end network obtained improved Internet performance. Can end-networks make do with much lesser routing flexibility and still achieve reasonable levels of Internet performance? Furthermore, do end-networks *require* support from special-purpose infrastructure deployed in the wide-area Internet? This dissertation provides direct answers to these questions, as we explain shortly.

2.3 Commercial Route Control Products

A number of vendors have recently developed "intelligent" end-network routing appliances or "route control" mechanisms. In contrast with overlay-based mechanisms which require an exclusive infrastructure to enable alternate routes, these routing appliances are purely end point-based. As such they are much easier to deploy and use. However, this very advantage also limits the amount routing flexibility these devices can provide in comparison with overlay-based approaches.

The basic premise in these approaches is that an end-network has Internet connections to multiple ISPs. This practice is referred to as *multihoming*. Given the multiple ISP connections, these commercial products allow a multihomed end-network to dynamically schedule their Internet traffic across its upstream ISP links for optimizing performance or availability. These devices are usually co-located with the end-networks edge routers, as shown in Figure 2.3. The products work by probing the performance of the end-network's ISP links to various destinations and scheduling traffic across the ISP that is most likely to offer the best performance or availability to a particular destination. Therefore, route control appliances can help subscriber networks avoid performance and availability problems in their ISPs' paths by dynamically routing via alternate ISPs.

Notice that each ISP provides the end-network with exactly one BGP path per destination, as explained in Chapter 1. In effect, with three ISP connections, a multihomed end-network has three paths per destination. Therefore, route control products help end-networks to make a clever choice of which ISP link, or, equivalently, which BGP path, to employ for obtaining the best possible performance or availability. While route control products provide more routing flexibility than traditional singly-homed BGP-based routing, this flexibility is nowhere near that offered by overlay networks. In this dissertation, we compare the relative trade-offs of the extents of flexibility pro-

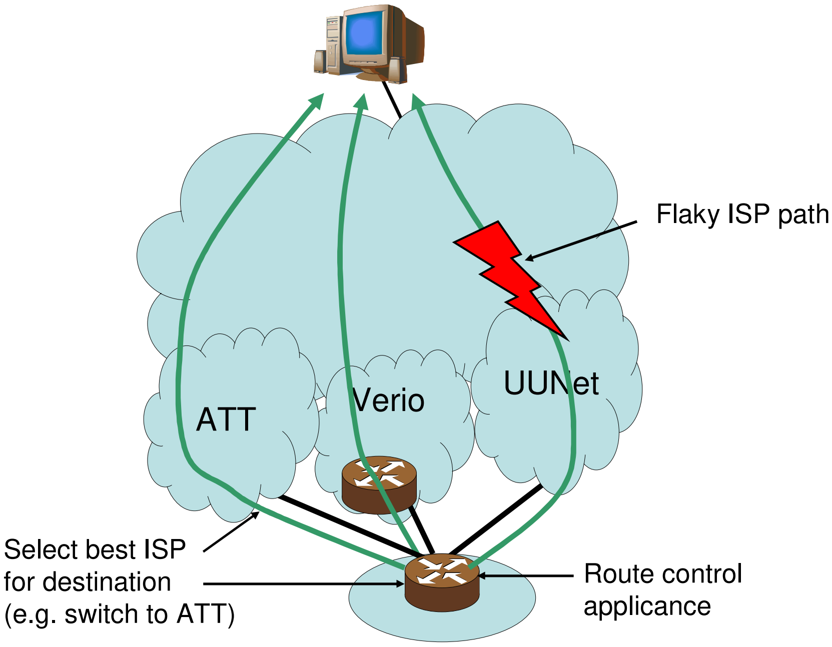


Figure 2.3: Multihoming route control: This figure illustrates a route control scenario for an end-network with three ISP connections.

vided by the two systems. We also study how to effectively utilize the modest amount of flexibility provided by route control to effectively improve Internet access performance.

While optimizing performance and availability (e.g. for Web transactions) is a primary design goal of these devices, they often provide other important benefits. For example, several route control devices allow end-networks to tune their traffic scheduling policies to obey specific bandwidth usage limits. Such appliances typically provide subscribers with a "knob" to trade-off improved performance for lower bandwidth usage costs (see, for example, [98, 103, 107]). Other appliances are targeted at balancing load across multiple, medium to low-speed ISP connections (e.g. multiple DSL links at home) to optimize Web download speeds [35, 94, 79]. A few others are specifically targeted at improving the responsiveness of delay-sensitive Internet applications such as VoIP [53, 111]. A detailed study of alternate applications of route control is beyond the scope of this dissertation.

2.4 An Overview of Our Approach

Most of what we know about the performance bottlenecks experienced by well-connected endpoints is driven by word-of-mouth. The common perception is that wide-area performance problems are confined to the boundaries between neighboring ISPs in the Internet, i.e., peering locations. Since no ISP has sufficient incentive to manage traffic at these locations, peering points are presumed to carry high-levels of traffic resulting in inefficient end-to-end performance.

Step #1: Performance bottlenecks in the current Internet [9] Chapter 3
Step #2: Multihoming route control for avoiding performance bottlenecks [7] Chapter 4
Step #3: Overlay routing versus multihoming route control [8] Chapter 5
Step #4: Practical multihoming route control strategies [10] Chapter 6
Step #5: Performance scaling in the future Internet [5] Chapter 7

Table 2.2: Overview of our approach: The five key problems central to this dissertation. We also show provide citations for the preliminary conference versions of the corresponding results.

The first issue we tackle in this dissertation is to investigate the verity behind this popular belief (Step #1 in Figure 2.2). Specifically, we seek to not only quantify the levels of traffic at peering locations, but also expose other locations of the wide-area Internet which could potentially limit the performance of well-connected endpoints. Apart from their location, we derive several other characteristics of the bottlenecks links, such as their latencies and available capacity.

We then explore techniques that well-connected end-networks may employ to overcome these performance problems. While many subscriber networks are beginning to employ route control appliances for this purpose, very little is known about the tangible benefits of these products. Subscriber networks often have to rely on word-of-mouth before deciding whether or not to employ such mechanisms. A moderately useful source of information for such subscribers is the accompanying set of "white papers" that outline the distinguishing features of the products. However, such documents merely help subscriber networks to select between products on the basis of which features are supported or not. The subscribers, in turn, will have to "try out" the appliances in order to understand their relative merits. Even in such situations, it is impossible for the subscribers to know if the Internet performance they observe from a specific route control appliance can be improved even further. The exact improvements from route control are also tightly coupled with the ISPs that the subscriber network connects to. For most end-networks, connection cost, and in some cases, the service level agreements (SLAs) offered by ISPs are the primary yard-sticks in determining whom to buy connectivity from. However, there is very little guidance available to subscriber networks in selecting ISPs that offer the expected level of performance, while also charging reasonable connectivity fees.

In this work, we seek to establish a base-line for the expected performance and resilience improvements from multihoming route control mechanisms (Step #2 in Figure 2.2). Further, we aim to investigate how end-networks should select their ISPs to realize the potential improvements from route control. To realize this goal, we collect performance and availability data over selected nodes in the server infrastructure of a large content distribution network operated by Akamai Technologies [2]: we call this collection of nodes our "measurement testbed". The measurement samples are obtained at small time intervals over several week-long intervals of time.

Apart from helping us understand the tangible benefits from route control, this approach offers three other important advantages:

1. Since Akamai has servers deployed at major cities throughout the world, our measurement study can help make observations about the improvements from route control across several of geographic regions of the world.

2. Akamai's server deployment style (i.e., several servers deployed across many different ISPs in major cities), helps us understand the impact of the number and the exact choice of ISPs on the expected performance improvements.

3. Since our measurements span long periods of time, we can derive hourly and diurnal patterns in the performance improvements from route control. This information can be employed to fine-tune the operation of route control appliances.

We also use the measurements above to contrast the improvements from route control with overlay routing-based mechanisms (Step #3 in Figure 2.2). Commercial overlay-based mechanisms offer end-networks a viable option for obtaining better Internet performance and resilience. Hence our comparison seeks to understand if there are clear benefits from overlay-based mechanisms that route control simply cannot provide. To evaluate the benefits of overlay routing, we employ the nodes in our testbed also as intermediate overlay hops. While our overlay testbed is not nearly of the same size as a commercial overlay network, it is nevertheless larger than other overlay testbeds employed in past research studies. We also evaluate a powerful form of overlay routing which combines the first-hop route choices that multihoming provides with the flexibility provided by overlay routing beyond the first hop ISP(s). The routes enabled by this form of overlay routing subsume the routes enabled by route control, and as such route control should offer strictly inferior performance. Our goal, then, is to understand how inferior route control really is.

Following the comparison of overlay routing and multihoming route control, we present a study of the effectiveness of simple route control strategies (Step #3 in Figure 2.2). Most route control white papers outline a few techniques that are believes to be key to the effectiveness of the respective products. We implement a few of these techniques and propose several new alternatives. In our evaluation, we highlight important drawbacks of existing approaches. We also seek to outline several other best common practices for the design and operation of route control appliances. The primary goal of our implementation effort, however, is to show that the simple route control schemes we develop can help end-networks significantly improve their Internet performance in practical deployment scenarios. A secondary goal is to show that the performance from these techniques is also reasonably close to the maximum benefits from route control.

While the first three steps in this dissertation deal with performance improvements in Internet today, Step #4 (See Figure 2.2) asks whether good Internet performance can be sustained in the future Internet. In a few years' time, the Internet will grow significantly in size. Furthermore, endpoint access speeds will improve by orders of magnitude. New killer applications will drastically alter the traffic mix on the network. Our goal, then, is to understand if the Internet is fundamentally capable of accommodating these future changes. If this is indeed the case, then we can expect that the techniques proposed for route control in this thesis, and other related mechanisms, will continue to remain effective at extracting good Internet performance in the future. However, it is possible that fundamental properties of the design and evolution of the Internet may prevent this from happening. Specifically, we investigate ask whether the Internet's interconnection topology and routing protocols together can limit the network's ability to accommodate key changes over time. We also identify guiding principles for the design and evolution of the network that can improve its ability to support robust future performance.

Chapter 3

An Empirical Evaluation of Wide-Area Internet Bottlenecks

It is widely believed that poor Internet performance arises primarily from constraints at the *edges* or the *access portions* of the network. These narrow-band access links (e.g., dial-up, DSL, etc.) limit the ability of applications to tap into the plentiful bandwidth and negligible queuing available in the interior of the network. As access technology evolves, enterprises and end-users, given enough resources, can increase the capacity of their Internet connections by upgrading their access links. Indeed, end-networks today often deploy access links with speeds in excess of 100Mbps. The positive impact on overall performance from employing higher access speeds may be insignificant, however, if other parts of the network subsequently become new performance bottlenecks. Ultimately, upgrades at the edges of the network may simply shift existing bottlenecks and hot-spots to the *non-access* or the *wide-area* portions of the Internet. To optimize the Internet access experience of end-users and end-networks, therefore, it is important to understand the location and traffic load of links in the wide-area Internet. In this chapter, we present a study of the likely location and characteristics of bottleneck links in wide-area Internet.

As noted in Chapter 2, the wide-area portion of the Internet is composed of several hundreds of ISPs, belonging to one of the four tiers of the Internet's ISP hierarchy. Several of these ISPs attach or "peer" with each other at various geographic locations. A *wide-area bottleneck*, therefore, could lie either entirely inside an ISP network or at peering locations. A more technical definition of wide-area bottlenecks follows:

> **Wide-area Bottlenecks** The wide-area bottleneck link on the path between two well-connected Internet endpoints is the link within an ISP (i.e., an *intra-ISP* link) or at the boundary between neighboring ISPs along the path (i.e., a *peering* link) that could potentially constrain the transfer speed of an *unconstrained TCP* flow between the endpoints.

An unconstrained TCP connection has the following two characteristics: (1) It does not suffer from access bandwidth or buffer size limitations at the sender or the receiver; and (2) When run for a sufficiently long period of time, the average transfer speed of the flow equals the average *available* *capacity* along the path. To clarify, the instantaneous available capacity of a network link is the raw bandwidth of the link less the current traffic volume across the link. The instantaneous available capacity of a path is the minimum of the available capacities of the links in the path. The average available capacity is simply the time-average of the instantaneous available capacity on the path.

Our primary objective is to devise a methodology for discovering and classifying wide-area bottleneck links according to their location in the Internet's ISP hierarchy—that is, identify the size and expanse of the ISP(s) that the bottleneck links belong in. A secondary objective is to derive other important properties of the bottleneck links. One such property is the *latency* of the bottleneck links, i.e., whether the typical bottlenecks are "short" link, confined to a city, or "long" cross-country links. Finally, we also derive the available capacity of the bottleneck links.

The main challenge in characterizing Internet bottlenecks is to measure paths that are representative of typical routes in the Internet, while avoiding biases due to a narrow view of the network from few probe sites. Further, we require probes which themselves are well-connected. Our results are based on measurements from 26 geographically diverse probe sites located primarily in the U.S., each with very high speed access to the Internet. We measure paths from these sites to a carefully chosen set of destinations, including paths to all Tier-1 ISPs, as well as paths to a fraction of Tier-2, Tier-3, and Tier-4 ISPs, resulting in 2028 paths in total. In addition, we identify and measure 466 paths passing through public Internet exchange points in order to explore the common perception that public exchanges are a major source of congestion in the Internet. A second challenge lies in actually measuring the bottleneck link and reporting its available bandwidth and location. Due to the need for control at both ends of the path, we were unable to leverage any of the existing tools to measure the available bandwidth. Hence, we develop a tool, *BFind*, which measures available capacity using a novel bandwidth probing technique.

We apply our measurement methodology to empirically determine the locations, estimated available bandwidth, and delay of non-access bottleneck links. Our results show that nearly half of the paths we measured have a non-access bottleneck link with available capacity less than 50 Mbps. Moreover, the percentage of observed paths with bottlenecks grows as we consider paths to destinations in smaller ISPs. Surprisingly, the bottlenecks identified are roughly equally split between intra-ISP links and peering links between ISPs. Also, we find that low-latency links, both within and between ISPs have a significant probability of constraining available bandwidth. Of the paths through public exchanges that had a bottleneck link, the constrained link appeared at the exchange point itself in nearly half the cases.

Chapter outline. Our measurement methodology, along with details on our choice of paths, and the design and validation of our measurement tool, BFind, is discussed in Section 3.1. In Section 3.2, we present our measurement-based characterization of wide-area bottleneck links. Finally, in Section 3.3, we discuss caveats of our approach, summarize the key observations in this chapter, and analyze the implications on end-user performance.

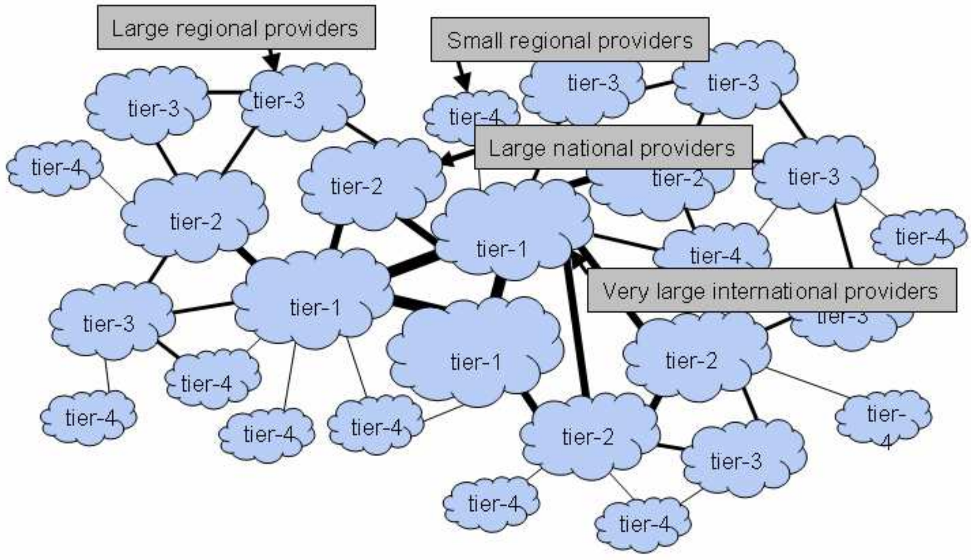


Figure 3.1: ISP hierarchy: This figure illustrates the four tiers in the Internet ISP hierarchy.

3.1 Measurement Methodology

The Internet today is composed of an interconnected collection of Autonomous Systems (ASes). These ASes can be roughly categorized as carrier ASes (e.g. ISPs) and stub ASes (end-customer domains). Our goal is to measure the characteristics of potential performance bottlenecks that well-connected end-nodes encounter that are not within their own control. To perform this measurement, we need to address three important issues:

1. Choosing an appropriate set of sources of measurement.
2. Choosing an appropriate set of probe destinations.
3. Developing a tool to identify and characterize bottlenecks along the probed paths.

In the first two cases, we need to pay careful attention to the ISPs that the sources and destinations are connected. To clarify our choice of measurement probe sites with sufficient diversity across ISPs, we briefly discuss the 4 tier classification of ISPs. Then, we present the details of our measurement probes and destinations, and our measurement too.

3.1.1 ISP Hierarchy

We use a hierarchical classification of ISPs or ASes into four *tiers* (as defined in [108]). The tier of an ISP provides an approximate indication of the size or the global reach of the ISP. For example, ISPs in tier-1 of the hierarchy, such as AT&T and Sprint, are large ISPs that do not have any upstream

ISPs. Most ISPs in tier-1 have peering arrangements with each other. Lower in the hierarchy, tier-2 ISPs, including Savvis, Time Warner Telecom and several large national ISPs, have peering agreements with a number of ISPs in tier-1. ISPs in tier-2 also have peering relationships with each other, however, they do not generally peer with any other ISPs. ISPs in tier-3, such as Southwestern Bell and Turkish Telecomm, are small regional ISPs that have a few customer ISPs and peer with a few other similar small ISPs. Finally, the ISPs in tier-4, for example rockynet.com, have very few customers and typically no peering relationships at all [108]. The tiers in the ISP hierarchy are illustrated in 3.1.

3.1.2 Choosing Traffic Sources

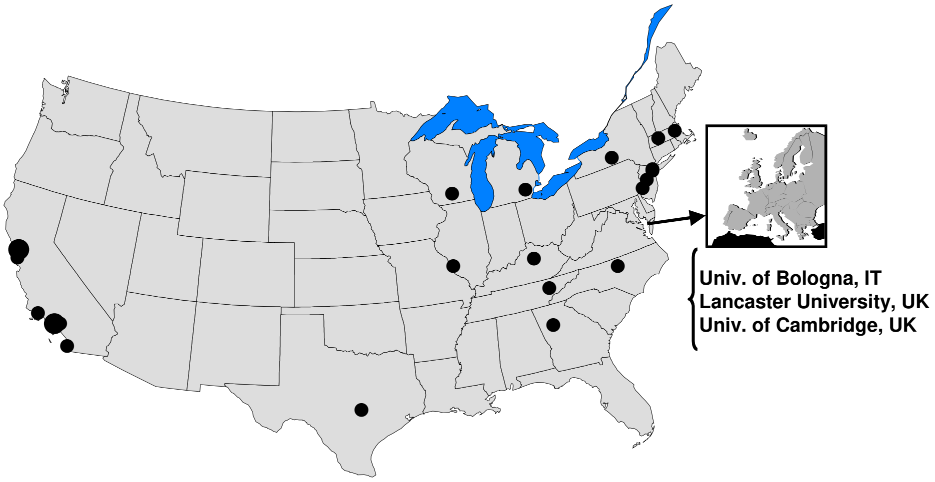
Stub ASes in the Internet are varied in size and connectivity to their provider networks. Large stubs, *e.g.* large universities and commercial organizations often have high speed links to all of their ISPs. Other stubs, *e.g.* small businesses, a much slower connection.

At the core of our measurements are traffic flows between a set of sources, which are under our control, and a set destinations which are random, but chosen so that we can measure typical Internet paths. Since we are interested in measuring bottlenecks faced by well-connected endpoints, we must ensure that the sources of our measurements have high-speed Internet connections as well. Large commercial and academic organizations are example of such endpoints. In addition, we must ensure that the sources are geographically diverse and span ISPs belonging to the four tiers. This ensures that the results are not biased by repeated measurement of a small class of bottlenecks links.

We use hosts participating in the PlanetLab project [88], which provides access to a large collection of Internet nodes that meet our requirements. PlanetLab is a Internet-wide testbed of multiple high-end machines located at geographically diverse locations. Most of the machines available at the time we performed our experiments (October 2002) were located in large academic institutions and research centers in the U.S. and Europe. Note that although our traffic sources are primarily at universities and research labs, we do not measure the paths *between* these nodes. Rather, our measured paths are chosen to be representative of typical Internet paths (e.g., as opposed to paths on Internet2).

Initially, we chose one machine from each of the PlanetLab sites as the initial candidate for our experiments. While it is generally true that the academic institutions and research labs hosting PlanetLab machines are well-connected to their upstream ISPs, we found that the machines themselves are often on low-speed local area networks. Out of the 38 PlanetLab sites operational at the outset of our experiments (October 2002), we identified 12 that had this drawback. We eliminated these 12 machines from the set of sources in our experiments.

The unique upstream ISPs and locations of the remaining 26 PlanetLab sites are shown in Figure 3.2. Specifically, as Figure 3.2(b) shows, the sources are connected to a diverse set of ISPs belonging to the 4 tiers of the ISP hierarchy.



(a) Source of measurement

	tier-1	*tier-2*	*tier-3*	*tier-4*
Total #unique ISPs	11	11	15	5
Avg. #ISPs per PlanetLab source	0.92	0.69	0.81	0.10

(b) First-hop connectivity of PlanetLab sources

Figure 3.2: Locations of PlanetLab sources (a) and their connectivity properties(b): Three of our sources and seven destinations are located in Europe (shown in the inset). The size of the dots is proportional to the number of sites mapped to the same location.

3.1.3 Choosing Probe Destinations

We have two objectives in choosing paths to measure from our sources. First, we want to choose a set of network paths that are representative of typical paths taken by Internet traffic. Second, we wish to explore the common impression that public network exchanges, or NAPs (network access points), are significant bottlenecks. Our choice of network paths to measure is equivalent to choosing a set of destinations in the wide-area as targets for our probe tools. Below, we describe the rationale and techniques for choosing test destinations to achieve these objectives.

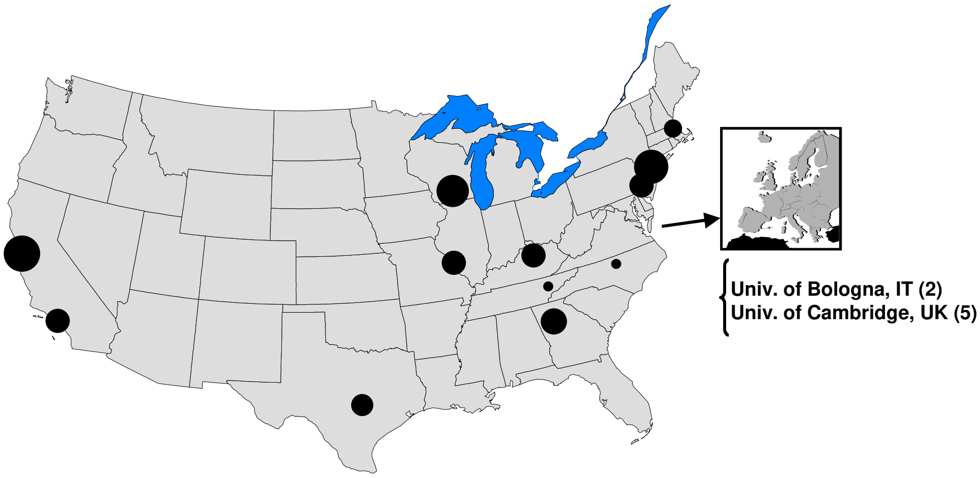
Most end-to-end data traffic in the Internet flows between stub networks. One way to measure typical paths would have been to select a large number of stub networks as destinations. However, the number of such destinations needed to characterize properties of representative paths would make the measurements impractical. Instead, we use key features of the routing structure of the Internet to choose a smaller set of destinations for our tests.

Traffic originated by a stub network subsequently traverses multiple intermediate autonomous systems before reaching the destination stub network. Following the definitions of AS hierarchy discussed earlier, flows originated by typical stub source networks usually enter a tier-4 or a higher tier ISP. Beyond this, the flow might cross a sequence of multiple links between ISPs and their higher-tier upstream ISPs (*uphill path*). At the end of this sequence, the flow might cross a single peering link between two peer ISPs after which it might traverse a *downhill path* of ASes in progressively lower tiers to the final destination, which is also usually a stub. This form of routing, arising out of BGP policies, is referred to as *valley-free* routing. We refer to the portion of the path taken by a flow that excludes links within the stub network at either end of the path, and the access links of either of the stub networks, as the *transit path*.

Clearly, non-access bottlenecks lie in the transit path to the destination stub network. Specifically, the bottleneck for any flow could lie either (1) *within* any one of the ISPs in the uphill or the downhill portion of the transit path or (2) *between* any two distinct ISPs in either portion of the transit path. Therefore, we believe that measuring the paths between our sources and a wide variety of different ISPs would provide a representative view of the bottlenecks that these sources encounter.

Due to the large number of ISPs, it is impractical to measure the paths between our sources and all such ISP networks. However, the *reachability* provided by these ISPs arises directly from their position in the AS hierarchy. Hence, it is more likely that a path will pass through one or two tier-1 ISPs than a lower tier ISP. Therefore, we test paths between our sources and *all* tier-1 ASes. Since tier-2 ISPs have lesser global reach, we only test the paths between our sources and a fraction of the tier-2 ISPs (chosen randomly). We measure an even smaller fraction of all tier-3 and tier-4 ISPs. The number of ISPs we chose in each tier is presented in Figure 3.3(b).

In addition to choosing a target AS, we need to choose a target IP address within the AS for our tests. For an ISP we choose, say <isp>, we pick a router that is a few (2-4) IP hops away from the server www.<isp>.com (or .net as the case maybe). We confirm this router to be *inside* the AS by manually inspecting the DNS name of the router where available. Most ISPs name their routers according to their function in the network, *e.g.* edge (chi-edge-08.inet.qwest.net) or backbone (sl-bb12-nyc-9-0.sprintlink.net), routers. The function of the router can also be inferred from the names of routers adjacent to it. In addition, we double check using the IP addresses of the ISP's routers along the path to www.<isp>.com (typically there is a change in the subnet address close to the web server). We measure the path between each of the sources and the above IP addresses. The geographic location of the destinations is shown in Figure 3.3(a). Each destination's location is identified by that of the traffic source with the least delay to it.



(a) Destinations probed (mapped to the closest source)

	tier-1	*tier-2*	*tier-3*	*tier-4*
Number tested	20	18	25	15
Total in the Internet [108]	20	129	897	971
Percentage tested	100	14	3	1.5

(b) Composition of the destination set

Figure 3.3: Location and connectivity the destinations: Each destination in (a) location is identified by the Planet-Lab source with minimum delay to the destination. Table (b) shows the composition of the destination set.

3.1.3.1 Public Exchanges

ISPs in the Internet peer with each other at a number of locations throughout the world. These peering arrangements can be roughly categorized as public exchanges, or NAPs, (e.g., the original 4 NSF exchanges) or private peering (between a pair of ISPs). One of the motivations for the deployment of private peering has been to avoid the perceived congestion of public exchanges. As part of our measurements, we are interested in exploring the accuracy of this perception. Therefore, we need a set of destinations to test paths through these exchanges.

We select a set of well-known NAPs, including Worldcom MAE-East, MAE-West, MAE-Central, SBC/Ameritech AADS and PAIX in Palo Alto. For each NAP, we gather a list of small, low-tier customers attached to the NAP. The customers are typically listed at the Web sites of the NAPs. Since these customers are low tier, there is a reasonable likelihood that a path to these customers from any source passes through the corresponding NAP (*i.e.*, they are not multihomed to the NAP and another ISP). We then find a small set of addresses from the address block of each of these customers that are reachable via traceroute. We use the complete BGP table dump from the Oregon route server [23] to obtain the address space information for these customers.

Next, we use a large set of public traceroute servers (153 traceroute sources from 71 ISPs) [113], and trace the paths from these servers to the addresses identified above using a script to automate finding and accessing working servers. For each NAP, we select all paths which appear to go through the NAP. For this purpose, we use the router DNS names as the determining factor. Specifically, we look for the name of the NAP to appear in the DNS name of any router in the path. From the selected paths, we pick out the routers one-hop away (both a predecessor and a successor) from the router identified to be at the NAP and collect their IP addresses. This gives us a collection of IP addresses for routers that could potentially be used as destinations to measure paths passing through NAPs.

However, we still have to ensure that the paths do in fact traverse the NAP. For this, we run traceroutes from the PlanetLab sources to the predecessor and successor IP addresses identified above. For each PlanetLab source, we record the subset of these IP addresses whose traceroute indicates a path through the corresponding NAP. The resulting collection of IP addresses is used as a destination set for the particular PlanetLab source.

3.1.4 Bottleneck Identification Tool: BFind

We now need a tool that we can run at the sources to measure the bottleneck link along the selected paths. As discussed earlier, the bottleneck link is the link along the path where the available bandwidth for an unconstrained TCP flow is the minimum[1]. We would like the tool to report the available bandwidth, latency and location (i.e. IP addresses of endpoints) of the bottleneck along a path. Before we describe our tool, we present a brief overview of existing tools and their drawbacks.

3.1.4.1 Exiting Tools

The development of tools to estimate the bandwidth characteristics of Internet paths continues to be an active research area (see [27] for a more complete list). Tools like *bprobe* [30], Nettimer [63], and PBM [87] use packet-pair like mechanisms to measure the *raw bottleneck capacity* along a path. Other tools like *clink* [32], *pathchar* [55], *pchar* [67], and *pipechar* [49], characterize hop-by-hop delay, raw capacity, and loss properties of Internet paths by observing the transmission behavior of different sized packets. A different set of tools, e.g., *pathload* [56], focus on the *available capacity* on a path. However, these tools require control over both the endpoints of the measurement. Since our destinations are routers not in our direct control, these tools are not applicable.

[1] Notice that a particular link being a bottleneck does not necessarily imply that the link is heavily utilized or congested.

TReno [70] usies UDP packets to measure available bulk transfer capacity. It operated in a single-ended mode and sends hop-limited UDP packets toward the destination. TReno emulates TCP congestion control by using sequence numbers contained in the ICMP error responses. It probes each hop along a path in turn for available capacity. Therefore, when used to identify bottlenecks along a path, TReno will likely consume ICMP processing resources for every probe packet at each router being probed as it progresses hop-by-hop. As a result, for high-speed links, TReno is highly intrusive. In what follows, we describe the design and operation of our bottleneck identification tool—BFind—that addresses the drawbacks of the above tools.

3.1.4.2 BFind Design

BFind's design is motivated by TCP's property of gradually filling up the available capacity based on feedback from the network. First, BFind obtains the propagation delay of each hop to the destination. Bfind uses the minimum of the (non-negative) measured delays along a hop as an estimate for the propagation delay on the hop[2]. The minimum is taken over delay samples from 5 traceroutes.

After this step, BFind starts a process that sends UDP traffic at a low sending rate (2 Mbps) to the destination. A trace process also starts running concurrently with the UDP process. The trace process repeatedly runs traceroutes to the destination. The hop-by-hop delays obtained by each of these traceroutes are combined with the raw propagation delay information (computed initially) to obtain rough estimates of the queue lengths on the path. The trace process infers that the queue on a particular hop is potentially increasing if, across 3 consecutive measurements, the queuing delay on the hop is greater than the maximum of 5ms and 20% of the raw propagation delay on the hop. This information, computed for each hop by the trace process, is constantly accessible to the UDP process. The UDP process then uses this information to constantly adjust its sending rate as described below.

If the feedback from the trace process indicates no increase in the queues along any hop, the UDP process increases its rate by 200 Kbps (the rate change occurs once per feedback event, *i.e.*, per traceroute). Essentially, BFind emulates the increase behavior of TCP, albeit more aggressively, while probing for available bandwidth. If, on the other hand, the trace process reports an increased delay on any hop(s), BFind flags the hop as being a potential bottleneck and the traceroutes continue monitoring the queues. In addition, the UDP process keeps the sending rate steady at the current value until one of the following events occur: (1) The hop continues to be flagged by BFind over *consecutive* measurements by the trace process and a threshold number (15) of such observations are made for the hop. (2) The hop has been flagged a threshold number of times in total (50). (3) BFind has run for a pre-defined maximum amount of total time (180 seconds). (4) The trace process reports that there is no queue build-up on any hop, implying that the increasing queues were only a

[2]If the difference in the delay to two consecutive routers along a path is negative, then the delay for the corresponding hop is assumed to be zero

transient occurrence.

In the first two cases, BFind quits and identifies the hop responsible for the tool quitting as being the bottleneck. In the third case, BFind quits without providing any reliable conclusion about bottlenecks along the path. In the fourth case, BFind continues to increase its sending rate at a steady pace in search of the bottleneck.

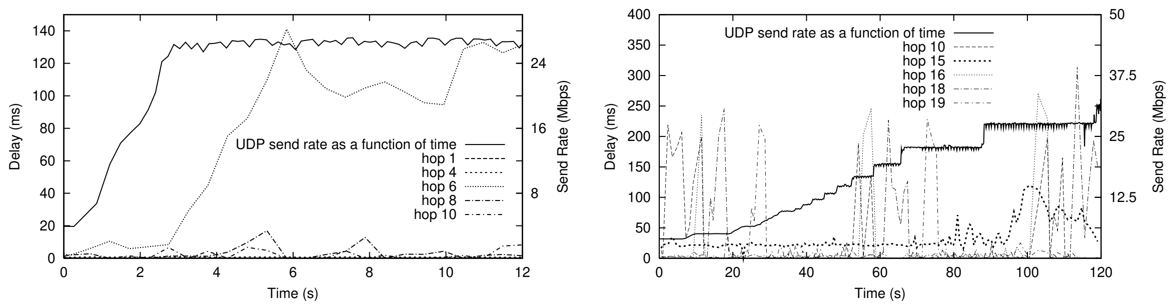
If the trace process observes that the queues on the first 1-3 hops from the source are building, it quits immediately, to avoid flooding the local network (The first 3 hops usually includes links in the source stub network). Also, we limit the maximum send rate of BFind to 50Mbps to make sure that BFind does not occupy too much of the local area network capacity. Hence, we only identify bottlenecks with < 50Mbps of available capacity. If BFind quits due to these exceptional conditions, it does not report any bottlenecks. Notice that BFind not only identifies the bottleneck link in a path, but also estimates the available capacity at the bottleneck. This equals the send rate just before the tool quit (upon identifying the bottleneck reliably). For paths on which no bottlenecks have been identified, BFind outputs a lower bound on the available capacity.

Notice that in several respects, the operation of BFind is similar to TCP Vegas's [25] rate-based congestion control. However, our sending rate modification is different than Vegas for two reasons. First, we want to ensure that the bottleneck link experiences a reasonable amount of queuing in order to come to a definitive conclusion. Therefore, BFind needs to be more aggressive than Vegas. Second, the feedback loop of the trace process is much slower than Vegas. One obvious drawback with this design is that BFind is a relatively heavy-weight tool that sends a large amount of data. This makes it difficult to find a large number of sites willing to host such experiments. BFind is not suitable for continuous monitoring of available bandwidth, but rather for short duration measurements. However, modifications to the design of BFind (see [50]) have addressed this issue.

3.1.4.3 BFind Operation: An Example

Figure 3.4 illustrates how BFind works. In Figure 3.4(a), BFind is run between planet1.scs.cs.nyu.edu (NYU) and r1-srp5-0.cst.hcvlny.cv.net (Cable Vision Corp, AS6128, tier-3). As BFind ramps up its transmission rate, the delay of hop 6 link between at-bb4-nyc-0-0-0-OC3.appliedtheory.net and jfk3-core5-s3-7.atlas.algx.net) begins to increase. BFind freezes its sending rate as the delay on this hop increases persistently. Finally, BFind identifies this hop as bottleneck with about 26Mbps of available capacity. This link also had a raw latency of under 0.5ms. The maximum queuing delay observed on this bottleneck link was about 140ms.

Figure 3.4(b) shows a potential false-positive. Run between planetlab1.lcs.mit.edu (MIT) and Amsterdam1.ripe.net (RIPE, tier-2), BFind observes the delays on various hops along the path increasing on a short time-scale causing BFind to freeze its UDP send rate quite often. The delay on hop 15 increases reasonably steadily starting at around 80 secs. This steady increase causes BFind to conclude that hop 15 was the bottleneck. However, it is possible that, similar to the other



(a) (b)

Figure 3.4: The operation of BFind: In either graph, queueing delay is shown on the left y-axis. The instantaneous UDP rate is shown on the right y-axis. In (a), BFind identifies hop 6 as the bottleneck. In (b), BFind identifies hop 15 as the bottleneck, although this could potentially be a false positive.

hops, this congestion was transient too, as indicated by a dip in the delay on hop 15 after 100secs. Therefore, we cannot entirely rule out the possibility of false-positives in our analysis. But we do believe that our choices of the set thresholds for BFind would keep the overall number of false positives reasonably low. We derive these threshold by testing various combinations that yielding minimum estimation error. Notice that false negatives might occur in BFind only when the path being explored was very free of congestion during the run, while being persistently overloaded at other times. Given that BFind runs for at least 30secs, and sometimes up to 150secs, we think that false negatives are unlikely.

3.1.4.4 BFind Validation

In this section, we first present simulation experiments in ns-2 [81] to address the following issues about our bottleneck estimation tool, BFind, presented in Section 3.1.4.2:

1. How accurately, and quickly, can BFind estimate the location of bottlenecks? Does the capacity of the bottleneck links or their location on the path impact the speed or accuracy? How does the presence of multiple bottlenecks affect the detection?

2. How does the bandwidth probing behavior of BFind compare with that of a TCP flow across the bottleneck link? Is BFind more or less aggressive than TCP?

3. How does BFind compete with long-lived TCP cross traffic while probing for available bandwidth at a bottleneck link (given that the bottleneck faced by the competing TCP flows is different from that faced by BFind)?

To address the above issues, we port BFind to ns-2. We setup path topologies shown in Figure 3.5(a) and (b). We chose not to experiment with more complicated topologies since BFind

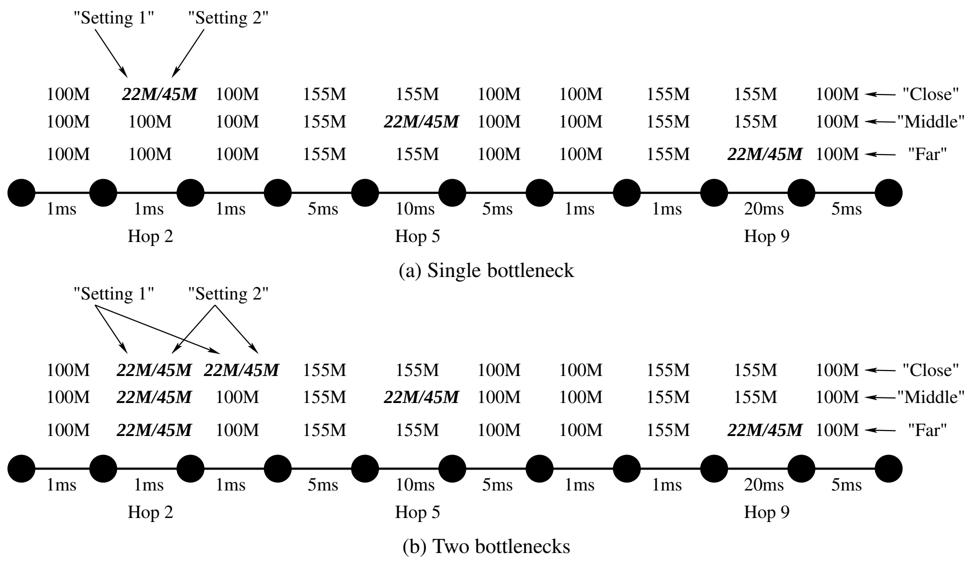


Figure 3.5: Topology used for BFind NS simulations: The topologies are explained in detail below. "M" stands for Mbps. The first row corresponds to location of the bottleneck link being "close", the second corresponds to "middle" and the third to "far".

probes only along a single path. As a result, all other nodes and links in the topology become auxiliary. In either figure, the path contains 10 hops (the hop-by-hop delays used we were observed on traceroutes from a machine in CMU and `www.amazon.com`). The capacity of the bottleneck links in the path is shown in italics.

In Figure 3.5(a), there is exactly one bottleneck in the path. To test the probing behavior of BFind we vary the location of this bottleneck link along the path (between "close", "middle" and "far", as explained below), as well as its raw capacity (between 22Mbps - referred to as "Setting 1" - and, 45Mbps - referred to as "Setting 2" shown in italicized bold font). In Figure 3.5(a), when the location of the bottleneck link is "close", hop 2 is the bottleneck link; when the location is "middle", hop 5 is the bottleneck; and when the location is "far", hop 9 is the bottleneck. Therefore, Figure 3.5(a) pictorially summarizes 6 different experiments with a single bottleneck link on the path – 3 different bottleneck "Locations", each with 2 different bandwidth "Settings".

The topology in Figure 3.5(b) has two similar bottleneck links. In "Setting 1", both links have an identical capacity of 22Mbps; in "Setting 2", they have an identical capacity of 45Mbps. When the "Location" of the bottlenecks is "close", hops 2 and 3 are chosen to be the identical bottleneck links; when it is "middle", hops 2 and 5 are the bottlenecks; and when it is "far", hops 2 and 9 are the bottlenecks.

In either topology, unless otherwise specified, there is cross traffic between neighboring routers. The cross traffic consists of 25 HTTP sessions in NS-2, each configured with 25 maximum connections. In addition, the cross traffic also includes 25 constant rate UDP flows with default parameters as set in NS-2. Cross traffic on the reverse path between neighboring routers is also similar. On

Location	Capacity of bottleneck link							
	Capacity = 22Mbps ("Setting 1")				Capacity = 45Mbps ("Setting 2")			
	BFind Output	Time	Available BW (BFind)	TCP Throughput	BFind Output	Time	Available BW (BFind)	TCP Throughput
Close	2	17.1	5.8	4.6	2	26.1	8.2	20.53
Middle	5	20.6	6.2	5.12	5	51.1	15.8	23.1
Far	9	19.6	6.2	4.5	9	57.1	20.2	24.09

Table 3.1: The bandwidth-probing performance of BFind: The table shows, for each of the six configurations of the topology in Figure 3.5(a), the output obtained from BFind and its comparison with a TCP flow on the bottleneck hop.

Location	Capacity of bottleneck link			
	Capacity = 22Mbps ("Setting 1")		Capacity = 45Mbps ("Setting 2")	
	BFind Output	Time	BFind Output	Time
Close ($2nd$ and $3rd$ hops)	2	17.1	2	17.1
Middle ($2nd$ and $5th$ hops)	5	22.1	2	29.6
Far ($2nd$ and $9th$ hops)	9	17.1	2	38.6

Table 3.2: Performance of BFind in the presence of two similar bottlenecks: The table shows the hops identified by BFind as being the bottleneck in each of the six configurations in Figure 3.5(b), and the time taken to reach the conclusion.

average, the cross traffic on all hops is similar.

In Table 3.1, we show the performance of BFind for the six experiments in Figure 3.5(a). We show if BFind correctly identifies the appropriate bottleneck, the time taken until detection, and the available bandwidth reported by BFind. In addition, we also report the average throughput of a TCP connection across the bottleneck link. The TCP connection runs under the *exact* same conditions of cross traffic as BFind. The results show that: (1) BFind accurately determines bottleneck links for both capacity values. When the capacity of the bottleneck link is higher, the time taken by BFind is not necessarily worse. (2) The throughput probed by BFind is roughly similar to that achieved by the TCP connection. When the capacity of the bottleneck link is low, BFind probes somewhat more aggressively than TCP; however, when the capacity is higher, BFind does not probe as aggressively.

In Table 3.2, we show the results for the performance of BFind in the presence of two, very similar, bottlenecks along a path (the topology in Figure 3.5(b)). The results show that BFind identifies one of the two links as being a bottleneck. However, the output is non-deterministic. To further investigate BFind's ability to detect bottlenecks where multiple such links may exist along a path, we slightly modified the topology in Figure 3.5(b) as follows: instead of two identical bottlenecks along the path, we deliberately set one of them to a *slightly* higher capacity. In "Setting

Location	Capacity of bottleneck link			
	Capacity = 22Mbps ("Setting 1")		Capacity = 45Mbps ("Setting 2")	
	BFind Output	Time	BFind Output	Time
Close ($2nd$ and $3rd$ hops)	3	20.6	2	46.6
Middle ($2nd$ and $5th$ hops)	2	17.1	2	20.1
Far ($2nd$ and $9th$ hops)	2	17.1	2	27.6

Table 3.3: Performance of BFind in the presence of two slightly different bottlenecks: The table shows the hops identified by BFind as being the bottleneck in each of the six configurations in Figure 3.5(b) when the bandwidth of one of the hops on the path is chosen to be slightly higher than that of the other.

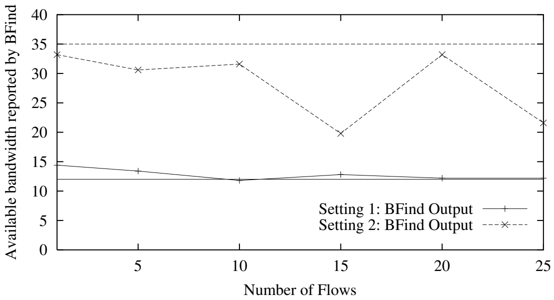


Figure 3.6: BFind interaction with competing long-lived TCP flows: The figures plots the available bandwidth reported by BFind for the topology in Settings 1 and 2, when competing long-lived TCP flows on the bottleneck hops are constrained to at most 10Mbps.

1", the second bottleneck link (i.e., hops 3, 5 and 9, respectively, in the three cases) now had a slightly higher capacity of 25Mbps. In "Setting 2", the capacity of the second link was chosen to be 50Mbps. The results for these experiments are shown in Table 3.3. In almost all cases, BFind correctly identifies hop 2 as the bottleneck link, despite the almost similar capacity of the second constrained link along the hop. Also, the time taken for detection is not necessarily worse.

We also show results demonstrating the interaction of BFind with competing long-lived TCP traffic. For these simulations, we use the single bottleneck topology in Figure 3.5(a), with a location of "mid" (i.e., bottleneck is hop 5). We eliminated cross traffic along hop 5 and, instead, started N long-lived TCP flows between router-4 and router-5 such that the total bandwidth achieved by the TCP flows is < 10Mbps at any point of time. We then started BFind on router 0 to probe for the available capacity on the bottleneck link, hop 5. Notice that in "Setting 1", BFind should report an available capacity of at most 12Mbps (since the raw capacity is 22Mbps), while in "Setting 2", it should report at most 35Mbps. In Figure 3.6, we plot the available bandwidth reported by BFind in either setting, as a function of the number of TCP flows, N. In "Setting 2", the bandwidth reported by BFind is always lower than 35Mbps, indicating that BFind does not have undesirable interactions with competing TCP traffic. In "Setting 1", the bandwidth from BFind is almost exactly 12Mbps

Destination Node	Path length	Pathload Report	Pipechar Report	BFind Report
CMU-PL	14	58.1 - 107.2Mbps	82.4Mbps	>39.1Mbps
Princeton-PL	12	91.3 - 96.8Mbps	94.5Mbps	>20.5Mbps
KU-PL	15	8.23 - 8.87Mbps	5.21Mbps (hop 12)	9.88Mbps (hop 12)
Pittsburgh-node	14	4.17 - 5.21Mbps	4.32Mbps (hop 11)	8.34Mbps (hop 11)
www.fnsi.net	11	N/A	8.2Mbps (hop 10)	8.43Mbps (hop 10)
www.i1.net	11	N/A	19.21Mbps (hop 7)	32.91Mbps (hop 8)

Table 3.4: BFind validation results: Statistics for the comparison between BFind, Pathload and Pipechar

as long as $N \geq 5$, again, show that BFind competes fairly with long-lived TCP traffic. (However, BFind is unfair in the RTT-proportional fairness sense).

Wide-Area Tests. Next, we present the results from a limited set of wide-area experiments to evaluate the available bandwidth estimate and the bottleneck location accuracy of BFind. To validate the available bandwidth estimate produced by BFind, we compare it against Pathload [56], a widely-used available bandwidth measurement tool. Pathload estimates the range of available bandwidth on the path between two given nodes. Since measurements are taken at either end of the path, control is necessary at both end-hosts.

To validate the bottleneck location estimation of BFind, we compare it with Pipechar [49], which operates similarly to tools like pathchar [55] and pchar [67]. Pipechar outputs the path characteristics from a given node to any arbitrary node in the Internet. For each hop on the path, Pipechar computes the raw capacity of the link, as well as an estimate of the available bandwidth and link utilization. We consider the hop identified as having the least available bandwidth to be the bottleneck link output by Pipechar and compare it with the link identified by BFind. We also compare the available bandwidth estimates output by BFind and Pipechar.

For these experiments, we perform transfers from a machine located at a commercial data center in Chicago, IL to a large collection of destinations. Some of these destinations are nodes in the PlanetLab infrastructure and hence we have control over both ends of the path when probing these destinations. The other destinations are random routers inside a few ISPs. When probing the path to the latter destinations, we do not have control over the destination end of the path. In total, we probe 30 destinations.

A sample of the results from our tests are presented in Table 3.4. The first three machines belong to the PlanetLab infrastructure. The fourth machine is located in Pittsburgh and attached via AT&T. The source is a host located in a Chicago area data center. The corresponding hop number for the bottleneck (if found) is also shown in parentheses. Note that since BFind limits its maximum sending rate it cannot identify bottlenecks with high available capacity (see the first example in Table 3.4). In the second case, the 180secs maximum execution time was insufficient for BFind to probe beyond 20Mbps[3]. The rest of the results show that BFind is reasonably consistently outputs the same location and available bandwidth as Pathload and Pipechar.

We also performed an initial cross-validation of our approach by checking if PlanetLab sources in a given metro area, attached to the same upstream ISP, identify the same bottleneck links when probing destination IP addresses selected in Section 3.1.3. For example, in the Los Angeles metro area, we found that the sources at UCSD, UCLA, UCSB, and ISI all identify similar bottlenecks in paths to the destinations in all cases where: (1) the bottlenecks are not located in their access network (CalREN2) and (2) the paths are identical beyond the access network.

3.1.5 Metrics Reported

In addition to the available bandwidth and the latency measurements, we post-process BFind's output to report on the ownership and location of Internet bottlenecks. We first classify bottlenecks based on *ownership*: bottlenecks either lie within ISPs, which we further classify by the tier of the owning ISP, or between ISPs, which we further classify according to the tiers of the ISPs at each end of the bottleneck. We identify the ISP owning the endpoint of any particular link using the whois servers from RADB [91] and RIPE [95] routing registries. Our second classification is based on the latency of the bottleneck links. We classify bottlenecks according to three different levels of latency – low latency ($<$ 5ms), medium (between 5 and 15ms) and high ($>$ 15ms). Though this is clearly a rough classification, we chose these classes to correspond to links at a PoP, links connecting smaller cities to larger PoPs, and long-haul links. For paths to the NAPs, we classify the path into three categories – those that do not have a bottleneck (as reported by BFind), those that have a bottleneck at the NAP, and those that have a bottleneck elsewhere. Again, we are only interested in non-access bottlenecks. Finally, we present a cumulative distribution of the available capacity of bottlenecks within a category.

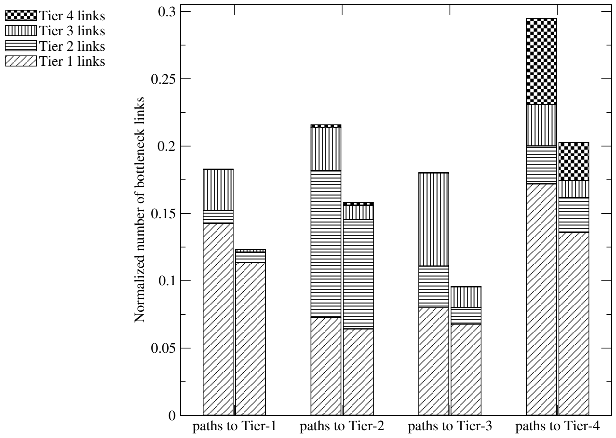
3.2 Results

Over a period of 5 weekdays, we ran BFind between the source and destination sites. The experiments were conducted between 9am and 5pm EST on weekdays. These tests identified a large number (889) of non-access bottleneck links along many (2028) paths.

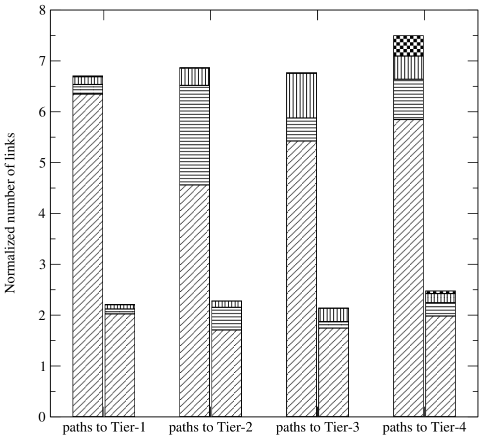
3.2.1 Path Properties

Before describing the properties bottleneck links, we present some important overall characteristics of the measured paths. Figures 3.7(b) and 3.8(b) summarize overall features of paths from PlanetLab

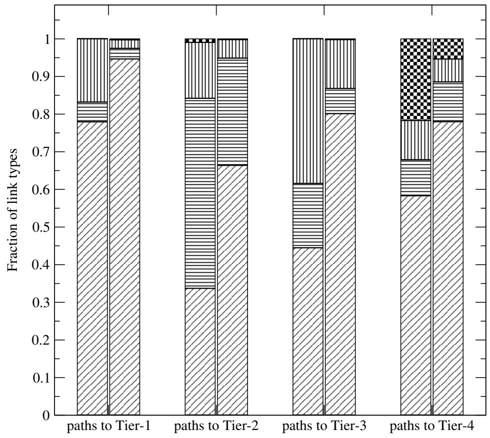
[3] In $>$ 97% of the paths we probed, BFind completed well before 180s, either because a bottleneck was found or because the limit on the send rate was reached.



(a) intra-ISP bottlenecks



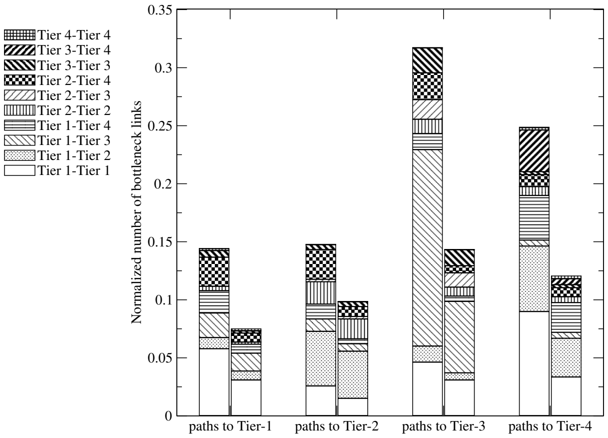
(b) All intra-ISP links



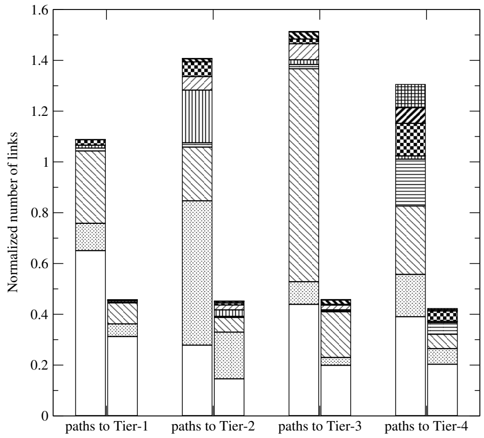
(c) Relative proportions: bottleneck links vs. all links

Figure 3.7: Relative prevalence of intra-ISP bottlenecks: Graph (a) shows the average number of bottlenecks of each kind appearing inside ISPs, classified by path type. The graph in (b) shows the total number of links (bottleneck or not) of each kind appearing in all the paths we considered. In (a) and (b), the left bar shows the overall average number of links, while the right shows the average number of unique links. Graph (c) shows the relative fraction of intra-ISP bottlenecks links of various types (left bar) and the average path composition of all links (right bar).

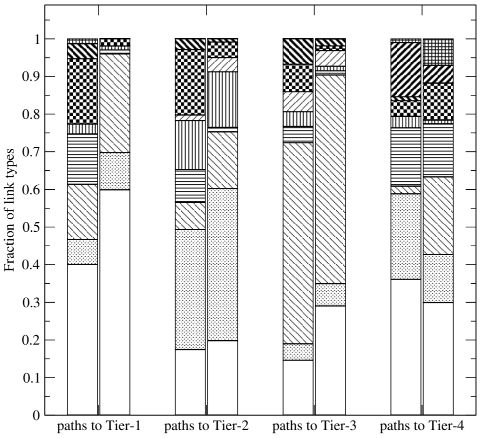
sites, classified by paths to ISPs of a particular tier. On the y-axis, we plot the normalized number of links, *i.e.*, the total number of links encountered of each type divided by the total number of paths in each class. Each path class has a pair of bars. While the left bars in the graphs show the overall average properties of the paths, the right bars show the average number of *unique* links that each path class adds to our measurements. This number is significantly less than the actual link counts (by a factor of 2 or 3). This is because links near the sources and destinations are probed by many paths (and are double-counted). Such links can bias our measurements since they may appear as



(a) Bottlenecks at peering links



(b) All peering links



(c) Relative proportions: bottlenecks vs. all links

Figure 3.8: Relative prevalence of peering bottlenecks: Graph (a) shows the average number of bottlenecks of each kind appearing between ISPs, classified by path type. The graph in (b) shows the total number of links (bottleneck or not) of each kind appearing in all the paths we considered. In (a) and (b), the left bar shows the overall average number of links, while the right shows the average number of unique links. Graph (c) shows relative fraction of peering bottlenecks of various types (left bar) and the average path composition for all links (right bar).

bottlenecks for many paths. Therefore, we also present information about unique links instead of describing average path properties alone.

Note that Figure 3.7 shows intra-ISP links while Figure 3.8 shows peering links. Characteristics of the entire paths are evident by examining the two together. For example, Figure 3.7(b) shows that the average path between a PlanetLab site and one of the tier-2 destinations traversed about 4.5 links inside tier-1 ISPs, 2.0 tier-2 ISP links, and 0.5 tier-3 links. Figure 3.8(b), which illustrates the location of the peering links, shows that these same paths also traversed about 0.25 tier-1 to tier-1 peering links, 0.75 tier-1 to tier-2 links, 0.2 tier-1 to tier-3 links, 0.2 tier-2 to tier-2 links, and a small number of other peering links. The total average path length of paths to tier-2 ISPs, then, is the sum of these two bars, i.e. $7 + 1.4 = 8.4$ hops. Similar bars for tier-1, tier-3 and tier-4 destinations show the breakdown for those paths. One clear trend is that the total path length for lower tier destinations is longer. The tier-1 average length is 7.8 hops, tier-2 is 8.3, tier-3 is 8.3 and tier-4 is 8.8. Another important feature is the number of different link types that make up typical paths in each class. As expected from the definition of the tiers, we see a much greater diversity (*i.e.,* hops from different tiers) in the paths to lower tier destinations. For example, paths to tier-4 destinations contain a significant proportion of all types of peering and intra-ISP links.

To summarize, we find that links involving tier-1 ISPs (intra-ISP links or peering links) form a significant fractions of the typical paths we measured. We also find that typical paths have far fewer peering links than intra-ISP links.

3.2.2 Locations of Bottlenecks

Figures 3.7(a) and 3.8(a) describe the different types of bottleneck links found on paths to destinations belonging to the 4 tiers. The left bars in the graphs show the probability that the identified bottleneck link is of a particular type, based on our observations. For example, from Figure 3.7(a), we see that the bottleneck links on paths to tier-2 networks consist of links inside tier-1 ISPs 7% of the time, tier-2 links 11% of the time, and tier 3 links 3% of the time (bottlenecks within tier-4 ISPs appear only in 0.2% of the cases). From Figure 3.8(a), we see that peering links of different types account for bottlenecks in tier-2 paths nearly 15% of the time, with tier-1 to tier-2 links appearing as the most likely among all types of peering bottleneck links. These two graphs together indicate that approximately 36% of tier-2 paths we measured had a bottleneck that we were able to identify. The other 64% appear to have bottlenecks with an available capacity greater than 50Mbps.

Figures 3.7(c) and 3.8(c) show the breakdown of links averaged across each type of path, for intra-ISP and peering links, respectively. Comparing the heights of components in the left and right bars gives an indication of the prevalence of the corresponding type of bottleneck link (left bar), relative to its overall appearance in the paths (right bar). From Figure 3.7(c), it first appears that lower-tier intra-ISP links are path bottlenecks in much greater proportion than their appearance in the paths. For example, Figure 3.7(c) shows that tier-3 links make up 17% of the bottlenecks to tier-1 destinations, but account for only about 2% of the links in these paths.

Note, however, that the right bars in Figure 3.7(a) show the number of unique bottlenecks links that we observed. Considering the first set of left and right bars (*i.e.*, all vs. unique bottlenecks for paths to tier-1 destinations) in Figure 3.7(a), we notice that there is a significant difference in the proportion of tier-3 bottleneck links. Upon further examination, we discovered that some of the PlanetLab sites were connected to the Internet via a tier-3 ISP. A few of the ISP's links were bottlenecks for many of the paths leaving the associated PlanetLab site. More generally, though, we see in Figure 3.7(c) that lower-tier intra-ISP links seem to be bottlenecks more frequently than we would expect based on the appearance of these links in the paths.

A similar examination of Figure 3.8(c) reveals several details about the properties of bottlenecks at peering links. Figure 3.8(c) shows that tier-1 to tier-1 peering links are bottlenecks less frequently than might be expected, given their proportion in the overall paths. Also, peering links to or from tier-2, tier-3 or tier-4 ISPs are bottlenecks more frequently than expected. For example, compare the proportion of tier-2 to tier-4 peering bottlenecks with the proportion of these links in the corresponding overall path length (e.g., 17% vs. 2% for paths to tier-1, and 17% vs. 4% for paths to tier-2).

Looking at Figures 3.7(a) and 3.8(a) together, we can observe some additional properties of bottleneck links. For example, total path lengths are around 8–9 hops (adding the heights of the bars in Figures 3.7(b) and 3.8(b)), of which only 1–1.5 hops are links between different ISPs. However, bottlenecks for these paths seem to be equally split between intra-ISP links and peering links (comparing the overall height of the bars in Figures 3.7(a) and 3.8(a)). This suggests that if there is a bottleneck link on a path, it is equally likely to be either in the interior of an ISP or between ISPs. Given that the number of peering links traversed is much smaller, however, the likelihood that the bottleneck is actually at one of the peering links is higher. But the fact that the bottleneck on any path is equally likely to lie either inside an ISP or between ISPs is surprising.

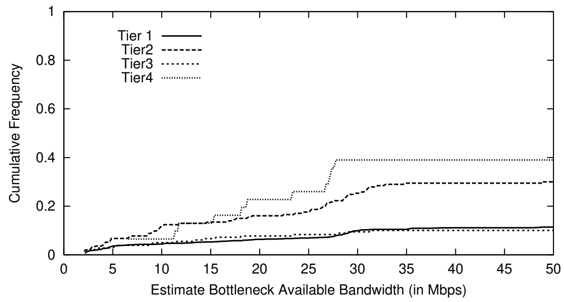
Another important trend is that the percentage of paths with an identified bottleneck link grows as we consider paths to lower-tier destinations. About 32.5% of the paths to tier-1 destinations have bottlenecks. For paths to tiers 2, 3, and 4, the percentages are 36%, 50%, and 54%, respectively. Note that while paths to tier-3 appear to have fewer intra-ISP bottlenecks than paths to tier-2, this may be because the peering links traversed on tier-3 paths introduce a greater constraint on available bandwidth.

To summarize, we found that wide-area bottlenecks are roughly equally split between intra-ISP and peering links. We also find that bottlenecks are more prevalent in networks lower in the ISP hierarchy.

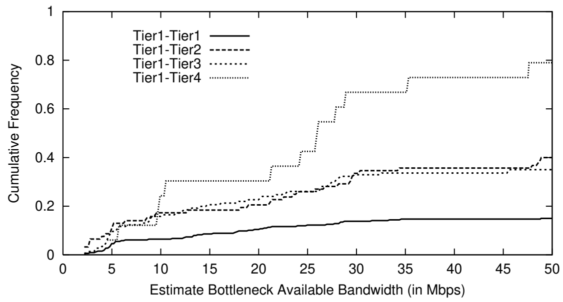
3.2.3 Bandwidth Characterization of Bottlenecks

In the previous section, we described the location and relative prevalence of observed bottleneck links. In this section, we discuss other properties of these bottlenecks. Specifically, we analyze the available bandwidth at these bottlenecks, as identified using BFind.

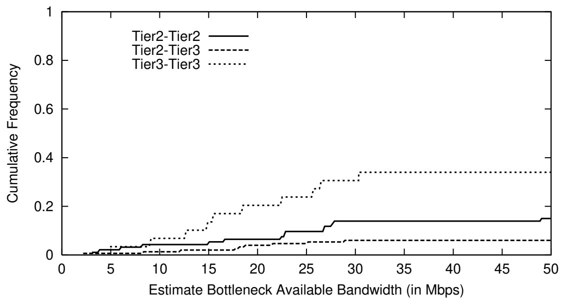
Figure 3.9 illustrates the distribution of available bandwidth of bottleneck links observed in different parts of the network. The CDFs do not go to 100% because many of the paths we traversed had more than 50 Mbps of available bandwidth. Recall that BFind is limited to measuring bottlenecks of at most 50 Mbps due to first hop network limitations. Figure 3.9(a) shows the bottleneck speeds we observed on intra-ISP links. The tier-1 and tier-3 ISP links appear to have a clear advantage in terms of bottleneck bandwidth over tier-2 ISP bottlenecks. The fact that the tier-3 bottlenecks



(a) intra-ISP



(b) Tier-1 peering



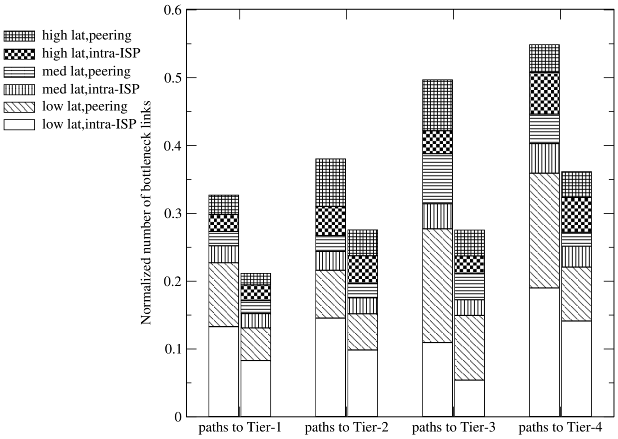
(c) Lower tier peering

Figure 3.9: Available capacity at bottleneck links: Graph (a) corresponds to bottlenecks within ISPs. Graphs (b) and (c) show the distribution of available capacity for bottlenecks in peering links involving Tier1 ISPs, and those in peering links not involving Tier1 ISPs, respectively. We do not show the distributions for bottleneck links between tiers 2 and 4 and those between tiers 3 and 4 since they were very small in number.

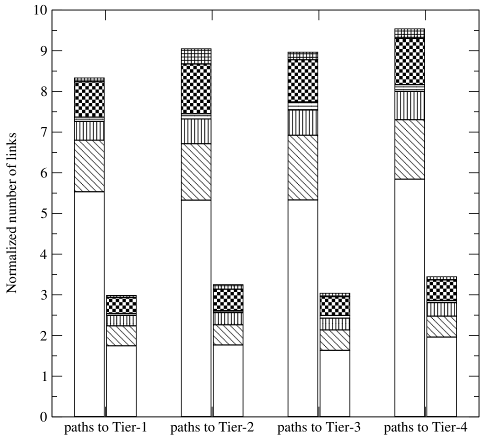
we identified offer higher available capacity than tier-2 bottlenecks was a surprising result. Links in tier-4 ISPs, on the other hand, exhibit the most limited available bandwidth distribution as expected.

In Figures 3.9(b) and (c) we consider the distribution of bottleneck bandwidth on peering links. Tier-1 to tier-1 peering links are the least constrained, indicating that links between the largest network providers are better provisioned when compared to links between lower-tier networks. Again, we find, surprisingly, that tier-2 and tier-3 links exhibit very similar characteristics, in their peering links to tier-1 networks (Figure 3.9(b)). Also, peering links between tier-2 and tier-3 are not significantly different than tier-2 to tier-2 links (Figure 3.9(b)). We do see, however, that bottleneck peering links involving networks low in the hierarchy provide significantly less available capacity, as expected. This is clearly illustrated in the bandwidth distributions for tier-1 to tier-4, and tier-3 to tier-3 links.

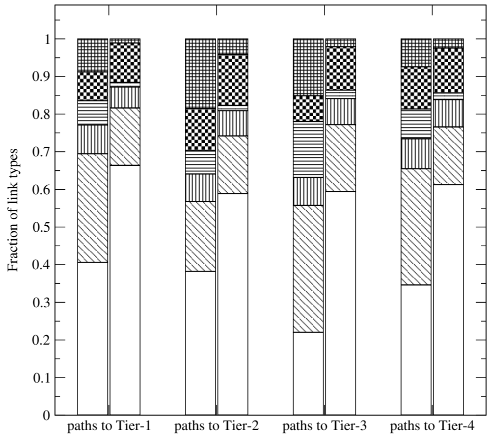
In summary, bottlenecks involving tier-1 ISPs seem to be fairly well-provisioned and have the highest available capacity. We also found that bottlenecks involving tier-3 ISPs had similar, if not better, available bandwidths than those involving tier-2 ISPs.



(a) Bottleneck Links



(b) All Links



(c) Relative proportions: bottlenecks vs. all links

Figure 3.10: Relative prevalence of bottlenecks of various latencies: Graph (a) shows the average number of bottlenecks of the three classes of latencies further classified into those occurring between ISPs and those occurring inside ISPs. Graph (b) shows the actual number of links (bottleneck or not) of each kind appearing in all the paths. Graph (c) shows the relative fraction of bottleneck links of various latency types (left bar) and the average path composition of all links (right bar).

3.2.4 Latency Characterization of Bottlenecks

In this section, we analyze the latency of bottlenecks. In particular, we explore the correlation between high-latency links and their relative likelihood of being bottlenecks. Figure 3.10 is similar to Figures 3.7 and 3.8. Figure 3.10(b) shows the overall latency characteristics of the paths. For example, paths to tier-2 destinations have an average of 5.3 low-latency intra-ISP, 1.4 low latency peering, 0.6 medium latency intra-ISP, 0.1 medium latency peering, 1.2 high latency intra-ISP, and 0.4 high latency peering links. In general, all path types have a high proportion of low-latency hops (both intra-ISP and peering) and high-latency intra-ISP hops. The latter is indicative of a single long-haul link on average in most of the paths we measured. While high latency peering links would seem unlikely, they do occur in practice. For example, one of the PlanetLab sites uses an ISP that does not have a PoP within its city. As a result, the link between the site and its ISP, which is characterized as a peering link, has a latency that exceeds 15ms.

In Figure 3.10(c) we illustrate the prevalence of bottlenecks according to their latency. We can observe that high-latency peering links are much more likely to be bottlenecks than their appearance in the paths would indicate. In observed paths to tier-2 destinations, for example, these links are 18.5% of all bottlenecks, yet they account for only 4% of the links. This suggests that whenever a high-latency peering link is encountered in a path, it is very likely to be a bottleneck. High latency intra-ISP links, on the other hand, are not overly likely to be bottlenecks (*e.g.*, 11% of bottlenecks, and 13.5% of overall hops on paths to tier-2).

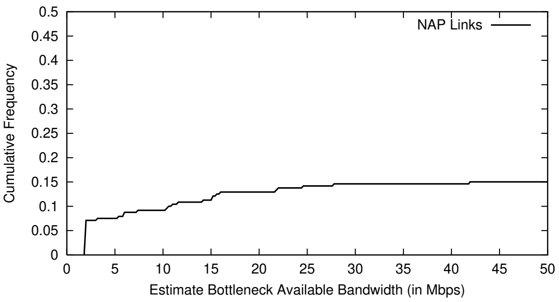
In general, Figure 3.10 suggests that peering links have a higher likelihood of being bottlenecks, consistent with our earlier results. This holds for low, medium, and high-latency peering links. For example, very few paths have any medium latency peering links, yet they account for a significant proportion of bottlenecks in all types of paths. Also, low-latency peering links on paths to the lower tiers (*i.e.*, tier-3 and tier-4) have a particularly high likelihood of being bottlenecks, when compared to paths to tier-1 and tier-2 destinations. Recall from Figures 3.9(b) and (c) that these lower-tier peering bottlenecks also have much less available bandwidth.

To summarize, we find that high-latency intra-ISP links are not overly likely to be bottlenecks. The opposite is true for high-latency peering links. We also find that peering links have a higher likelihood of appearing as bottlenecks on wide-area paths.

3.2.5 Bottlenecks at Public Exchange Points

We now present a study of bottlenecks along paths through public exchanges. As indicated in Figure 3.11(a), we tested 466 paths through public exchange points. Of the measured paths, 170 (36.5%) had a bottleneck link. Of these, only 70 bottlenecks (15% overall) were at the exchange point. This is in contrast to the expectation that many exchange point bottlenecks would be identified on such paths. It is interesting to consider, however, that the probability that the bottleneck link is located at the exchange is about 41% ($= 70/170$). In contrast, Figures 3.7(a) and 3.8(a) do not show any other type of link (intra-ISP or peering) responsible for a larger percentage of bottlenecks.[4]

#Paths to exchange points		466
#Paths with non-access bottlenecks		170
#Bottlenecks at exchange point		70



(a) Relative prevalence (b) Available bandwidth distribution

Figure 3.11: Bottlenecks in paths to exchange points: Table (a) on the left shows the relative prevalence of bottleneck links at the exchange points. Figure (b) shows the distribution of the available capacity for bottleneck links at the exchange points.

3.3 Measurement Caveats, Summary of Observations and their Implications

The key results from our study are summarized in Table 3.5. This study yields a number of interesting and unexpected findings about the characteristics of wide-area bottleneck links. For example, we find a substantial number of bottleneck links *within* ISPs (nearly 50% of the bottlenecks we found were located inside ISP networks). However, the conventional wisdom has been that most wide-area bottlenecks are confined to peering locations since there is little or no economic incentive for ISPs to carefully manage the load on links they share with their neighbors. In addition, we also observed that low latency links, whether within ISPs or between them, can also constrain available bandwidth with a small, yet, significant probability.

Furthermore, our observations can provide some guidance when considering issues such as choosing an access ISP and optimizing routes through the network. In what follows, we discuss some of these issues in the context of our empirical findings. First, however, we discuss some of the caveats of our measurement methodology.

3.3.1 A Critique of Our Measurement Methodology

We describe some possible shortcomings of our approach here. To approximate the measurement of "typical" paths, we choose what we believe to be a representative set of network paths. While

[4]However, in Figure 3.8(a), bottlenecks between tiers 1 and 3 in paths to tier-3 destinations are comparable to bottlenecks at exchange points in this respect.

Non-access bottlenecks are equally likely to be links within ISPs or peering links between ISPs.
The likelihood of the existence of a bottleneck increases on paths to lower tier ISPs.
Internal links in lower tier ISPs appear as bottlenecks with greater frequency than their overall presence in typical paths.
High-latency peering links are very likely to be the bottlenecks along the paths they appear in. High-latency intra-ISP links, on the other hand are not overly likely to be bottlenecks.
Interior and peering bottlenecks in tier-2 and tier-3 ISPs exhibit very similar available capacity.
hen a bottleneck is found on paths through a public Internet exchange, the likelihood of bottleneck actually lying at the exchange is more than 40%.
All paths have a high proportion of low-latency links (interior and peering) and roughly one high-latency interior link.

Table 3.5: Properties of wide-area Internet Bottlenecks: Summary of key observations regarding wide-area bottlenecks.

the set of paths is not exhaustive, we believe that they are diverse in their location and network connectivity. However, our test nodes are relatively USA-centric (only 3 international sources and 7 destinations) and do not measure international network connectivity well.

Route changes could also have a significant impact on our measurements. If an Internet route changes frequently, it becomes difficult for BFind to saturate a path and detect a bottleneck. Similarly, if an AS uses multipath routing, BFind's UDP probe traffic and its traceroutes may take different paths through the network. As a result, BFind may not detect any queuing delays despite saturating the network with traffic. If either of these situations occurred, traceroutes along the tested path would likely reveal multiple possible routes. However, despite our continuous sampling of the path with traceroute during a BFind test, we did not observe either of these routing problems occurring frequently. This is consistent with recent results showing that most Internet paths tend to be stable, even on an hour's timescale [119].

The processing time taken by routers to generate traceroute ICMP responses can impact our measurement of queuing delay and, therefore, bottlenecks in the network. Many researchers have noted that ICMP error processing, typically done in the router "slow" processing path, takes much longer than packet forwarding. In addition, some routers pace their ICMP processing in order to avoid being overwhelmed. Either of these could cause the delays reported by traceroute to be artificially inflated. However, recent work [46] has shown that slow path/fast path differences should not affect traffic measurement tools in practice since the typically observed ICMP processing delays are on the order of 1-2 ms, well within the timescales we need for accurate bottleneck detection.

Address allocation may also skew our results. We rely on using the address reported by routers in their response to traceroute probes to determine their ownership. However, in some peering arrangements, a router owned by an ISP is allocated an address from the peer ISP's address space to make configuration convenient. In such situations, our link classification may erroneously identify the incorrect link (by one hop) as a the peering link between the ISPs. However, we believe that the common use of point-to-point links in private peering situations and separate address allocations used in public exchanges (these both eliminate the above problem) reduce the occurrence of this problem significantly.

Finally, we note that our results represent an empirical snapshot of non-access Internet bottlenecks. That is, we focus on collecting observations from a large number of paths, rather than taking repeated measurements of a few paths over an extended period. While our approach provides a wider view of the characteristics and locations of bottlenecks, we cannot judge, for example, how stable or persistent the locations are.

3.3.2 ISPs and Provisioning

Our measurements show that there is a clear performance advantage to using a tier-1 ISP. Our results also show that small regional ISPs, e.g., tier-4 providers, have relatively low-speed connectivity to their upstream ISP, irrespective of the ISP's size. In addition, their networks often exhibit performance bottlenecks. This may be considered a reflection of the impact of economics on network provisioning, if we assume that ISPs lower in the AS hierarchy are less inclined to over-provision their networks if typical customer traffic volume does not thus far require it. As a result, there is a clear disadvantage to using a tier-4 ISP for high-speed connectivity. However, the trade-offs between tier-2 and tier-3 networks are much less clear.

Paths to tier-3 destinations had a larger percentage of bottleneck links than tier-2 paths. Despite this, we also observed that tier-2 and tier-3 bottlenecks show similar characteristics in terms of available capacity, with tier-3 bottlenecks (both intra-AS and peering links) performing slightly better in some cases. This might be explained if we conjecture that tier-2 ASes, by virtue of their higher degree of reachability, carry a larger volume of traffic relative to their capacity, when compared with tier-3 ASes. Extending this hypothesis, we might conclude that if a stub network desires reasonably wide-spread connectivity from its ISP, then choosing a tier-3 ISP might be a beneficial choice, both economically and in terms of performance, assuming that connectivity to tier-3 ISPs is less expensive.

3.3.3 Route Selection for Improved Internet Performance

Our measurements show that a large fraction of Internet paths (nearly 50% in our measurements) suffer from performance bottlenecks. However, the Internet has a very rich topology. Numerous alternate paths with a reasonable amount of available bandwidth do exist between arbitrary endpoints. Indeed, the remaining 50% of the paths we measured had an available capacity of 40-50 Mbps or more. This is true across most non-access links irrespective of their location or latency.

The observations in this chapter naturally lead to the following key question—Is it possible to avoid these wide-area bottlenecks, and improve Internet performance, by leveraging existing routing protocols? While there has been considerable work on load-sensitive routing of traffic within an AS, little is known about how to extend this across ASes. We address this question in the next two chapters.

Chapter 4

A Measurement-Based Analysis of Multihoming

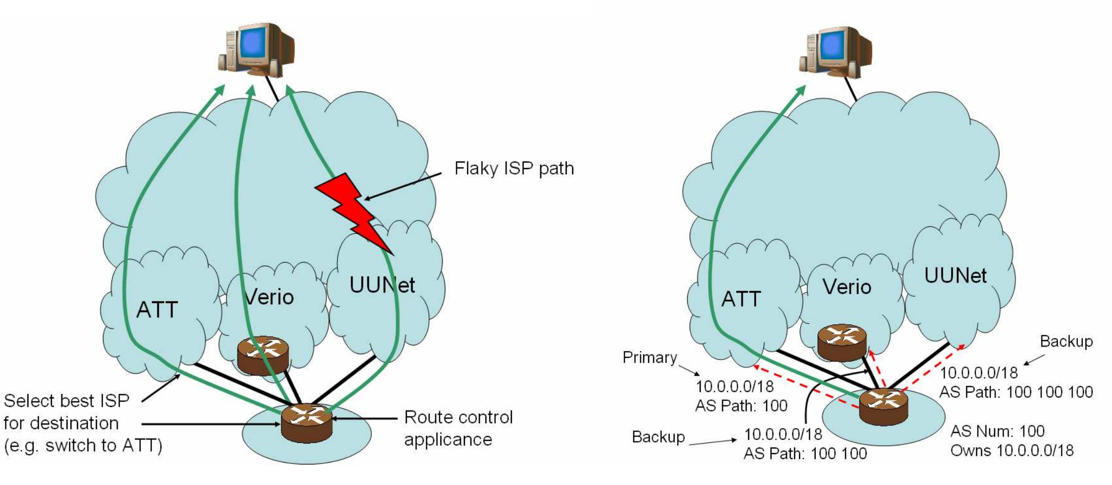
The Internet is constituted by several hundreds of distinct ISPs. In most large cities of the world, several tens of ISPs compete to provide Internet connectivity options to home users and businesses alike. Apart from pricing and offered connectivity speeds, these ISPs differ from each other in one crucial aspect: The performance and connectivity provided by the ISPs to different parts of the Internet vary substantially with time, the destination involved, and the choice of the ISP itself. For example, due to variations in traffic volumes in their networks (or in their neighboring ISPs' networks), or due to traffic engineering, ISPs may differ in the performance they offer to various Internet destinations over time. Similarly, depending on which ISPs Internet destinations are connected to, it may be better to use one ISP versus another to communicate with a specific destination.

As we mentioned in Chapter 1, when an end-network, such as an enterprise, a content provider or a university buys Internet connectivity from a single ISP, it is restricted to using the paths provided by the ISP to various Internet destinations. In general, the ISP provides exactly one BGP path per destination. Therefore, if the ISP, or networks further upstream, face performance bottlenecks, or availability problems, the end-network may receive poor download speed and response times, or even stay disconnected from key Internet destinations, for extended periods of time.

However, when the end-network subscribes to 2 or 3 different ISPs, an approach popularly referred to as multihoming, it enjoys a modest improvement in the number of available routes per destination. Indeed, each ISP provides one BGP path, yielding 2 or 3 almost distinct paths, per destination. The end-network can then intelligently switch between the ISPs, for example, based on the time of the day, or the Internet destination, while always using the best-performing ISP for a given transfer. Such an approach to Internet connectivity could significantly improve the Internet performance and reliability of the subscriber networks. In general, we will refer to this approach as *multihoming route control* (illustrated in Figure 4.1(a)). A more technical definition follows:

> **Multihoming Route Control** is the technique in which an end-network buys connections from multiple ISPs and intelligently schedules its traffic across the ISP connections, so as to improve Internet transfer speeds, response times and reliability.

Multihoming has been traditionally employed by large end-networks to achieve "coarse-grained"



(a) Route control

(b) Primary-backup mode

Figure 4.1: Multihoming: Figure (a) shows an example of a route control set-up. Figure (b) shows a traditional multihoming set-up.

resilience from Internet service interruptions. In such a set-up, the multiple ISP connections are employed in a "primary-backup" mode, where all traffic is moved over to an available ISP link upon failure of the primary link (see Figure 4.1(b)). However, the advent and the expected growth of *route control* devices promises to allow subscribers to leverage other benefits from multihoming. For example, route control products can now be leveraged to optimize Web performance, transfer speeds, bandwidth and even availability (on very fine time-scales) across multiple ISP links. Aside from marking statements, however, little is known about the tangible benefits end-networks can expect from such products and services.

In this chapter, our goal is to quantify the extent to which subscriber end-networks can benefit from employing multihoming route control mechanisms. We focus both on improvements in Internet performance as well as reliability. Conceptually, our approach is to consider a network subscriber in a major metropolitan area, and evaluate the relative benefits of choosing upstream ISPs from several available options. We assume that the subscriber has little or no control over end-to-end paths, but rather only which ISPs provide first-hop connectivity to the Internet, and how the subscriber's traffic is scheduled across the ISPs.

We collect and analyze several datasets in a step-by-step approach to quantify the *performance* benefits of route control[1]. First, we evaluate the improvements in Internet round-trip time (RTT) performance from a form of multihoming in which the subscriber has very coarse-grained control over the ISP links used for data transmission. For example, the subscriber may have to use a single ISP for all traffic, over an hour's period. We refer to this as *naive multihoming*. Then, we illustrate

[1]Here, and in the rest of the thesis, we use the terms multihoming, route control and multihoming route control interchangeably.

the improvements in RTT performance from per-connection control over traffic in a set-up where the subscriber has connections to two ISPs. We refer to this as *2-multihoming*. Finally, we analyze the more general notion of k*-multihoming*, $k \geq 2$, where the subscriber is multihomed to k available ISPs in a given city. We establish a baseline in which we assume that it is possible for a subscriber to employ all k ISPs and switch to the best performing ISP at each instant. By evaluating the performance as k is increased, we provide insight into the incremental benefit from additional ISPs. We quantify both the RTT as well as throughput improvements from k-multihoming. In addition, we analyze the impact of the choice of ISPs and the impact of the time of the day and day of the week on the benefits from k-multihoming. To quantify the reliability benefits of k-multihoming, we collect two further datasets. The first dataset quantifies the availability provided by multihoming based on active probe measurements. We use the second dataset to show how multihoming improves the availability of Internet paths using estimates of availability of Internet routers.

Most of our data sets comprise measurements conducted over the servers and monitoring nodes deployed by Akamai [2], a large content distribution service provider. These servers and monitors are attached to a diverse set of ISPs (most nodes connected to a single ISP), with multiple Akamai servers located many major metropolitan areas. The network performance data collected at these Akamai nodes allows us to compare performance across ISPs from the perspectives of multihomed subscribers in different metropolitan areas.

In order to compute the potential improvements from multihoming route control using the above datasets, we shall make three key assumptions in this chapter:

- The end-network has perfect information about the performance and availability of routes via each of its ISPs, whenever necessary.
- The end-network can switch between candidate ISPs to a destination as often as desired.
- The end-network can easily control the ISP link traversed by packets destined for its network. We refer to as "inbound control".

The key observation from our measurements is that, on average, multihoming can considerably improve performance and reliability of end-networks. In terms of performance, for example, even in a 2-multihoming situation, RTTs were improved by 25% on average for 3 out of 4 metro areas we study. The improvements in throughput performance were $\sim 20\%$ on average. We also find strong evidence of diminishing incremental performance benefits as more ISPs are added. We observe that increasing beyond $k = 3$ provides little added performance or reliability. Comparing the optimal multihoming solution to a random choice of k ISPs (for $k \leq 3$), we find that random selection degrades RTT performance by as much as 50%. This suggests that a careful choice of ISPs is key to achieving the potential benefits of multihoming.

Chapter outline. We illustrate the performance improvements from naive multihoming in Section 4.1. In Section 4.2, we illustrate the improvements from per-connection control of traffic with two ISP connections (2-multihoming). In Section 4.3, we study the improvements due to k-multihoming. We present the reliability analysis in Section 4.4. Finally, we summarize the observations from this study, and discuss their implications, in Section 4.5.

4.1 Naive Multihoming: RTT Performance

In this section, we study the improvements in Internet RTT performance when subscribers employ *naive multihoming*. Ideally, to realize the optimal benefits from route control, a subscriber may have to exercise per-flow control over the ISP link used (Per-byte control would, in theory, be even better. But this form of control may have poor interactions with current TCP implementations). In contrast, in naive multihoming, all of the subscriber's traffic uses a single ISP connection for a certain interval of time (e.g., a few hours), irrespective of the destination. At the end of the interval, the subscriber may decide to use a different ISP link for the upcoming interval.

4.1.1 Measurement Dataset

In our analysis of naive multihoming, we use passive measurement data collected at server nodes deployed by Akamai across various US cities. An important feature of this data is that the collection points are connected to a large variety of ISPs. Moreover, there are multiple metropolitan areas in which three or more collection points are located, each connected to a different ISP. We use such Akamai servers in a single metro area as stand-ins for a multihomed subscriber. We use the term *Multihoming Emulation* to refer to this procedure.

The dataset we employ contains the average HTTP *turnaround times* for requests made by Akamai servers back to various origin content provider servers (Figure 4.2). These requests are typically initiated when an Akamai server does not have a valid object cached and has to retrieve it from the origin server. The turn-around time for such HTTP requests is the time between the transfer of the last byte of the request from the Akamai node and the receipt of the first byte of the response from the origin server. Hence, the turnaround time offers a reasonable estimate of network delay. Since the customer content providers of the CDN are large Web servers, we expect their servers to be well-provisioned, and therefore the observed turnaround time should be constituted mainly by network delay with almost no delay due to the Web server itself.

The turnaround times we measure are averaged every hour across requests sent to various origin content providers. We collected this data for each hour over two five-day periods: Monday, 6^{th} January 2003 to Friday, 10^{th} January 2003 and Monday, 13^{th} January 2003 to Friday, 17^{th} January, 2003 (both inclusive).

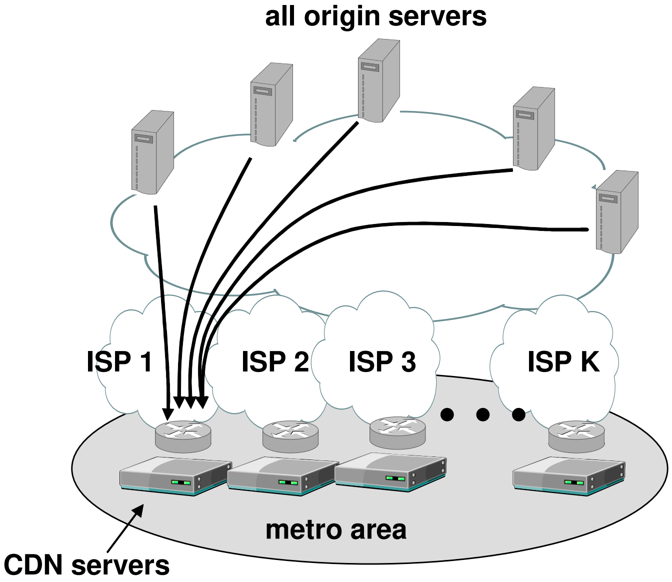


Figure 4.2: Naive Multihoming: Akamai servers connected to different ISPs in the same city download objects from all customer origin servers in order to serve them to clients. For this data set, turnaround times are averaged over each hour across retrievals from various origin servers.

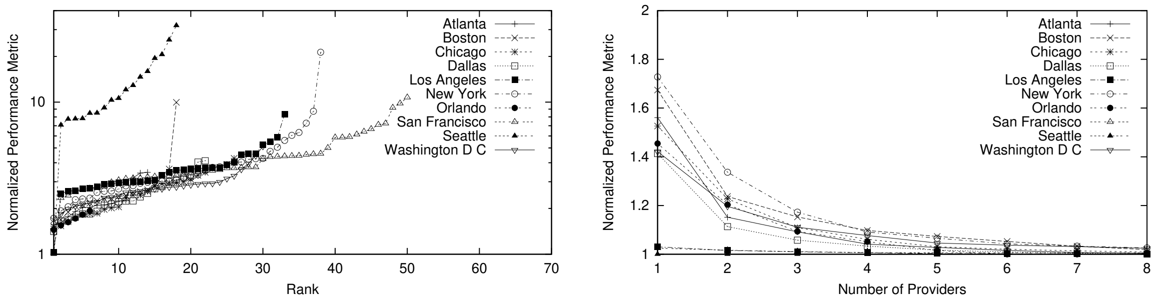
4.1.2 Measurement Results

We compute the performance improvements from naive multihoming as follows. For each hour, we compare the average turnaround time achieved by using the best ISP among all those available in the city, with that from using the best ISP in a candidate multihoming option (i.e., a given subset of ISPs). We average this ratio over all hours, and report the minimum normalized performance metric (the minimum is taken over all possible candidate options).

Formally, we compute the performance benefits from a given naive multihoming option consisting of k ISPs (denoted by OP_k) as follows:

$$N_{OP_k} = \frac{\sum_t (HT_{OP_k}(t)/HT_{best}(t))}{Numvalid(t)}$$

where N_{OP_k} is the performance of using the k-multihoming option OP_k in a given city, relative to the performance of using all available ISPs. $HT_{OP_k}(t)$ denotes the best average turnaround time performance among the k ISPs in the set OP_k at hour t. $HT_{best}(t)$ is the best average turnaround time performance at hour t over all the available carriers. The sum in the numerator is taken over all hours t for which all the k ISPs have the average turnaround time statistics logged in the data set $\mathcal{H}_1$. $Numvalid(t)$ counts the number of such instances t. For a very small fraction of the hours, the average turnaround time data was unavailable for certain networks.



(a) Ranks of ISPs

(b) Naive k-multihoming relative benefits

Figure 4.3: Naive k-multihoming: Figure (a) shows the 1-multihoming performance of the ISPs in each city, with ISPs ranked according to their performance. Figure (b) shows the diminishing returns from k-multihoming in each city.

In Figure 4.3(a) we plot the performance metric N_{OP_1} for ISPs in a city against their ranks (The ISP with rank 1 is the best in the city). The graph shows the first week of data; the second week is very similar. Notice that in some of the cities, there are a few ISPs that give significantly better performance than the others. For example, the best ISP in Seattle provides at least 7 times better performance as any other ISP. There are also cities in which many ISPs provide similar performance.

From Figure 4.3(a) it is apparent that, in some cities, there were in excess of 50 ISPs (e.g., San Francisco). Evaluating all the $\binom{50}{k}$ options for k-multihoming to determine the best naive k-multihoming option is computationally expensive. We reduce the amount of computation by evaluating k-multihoming options against the performance of up to 20 top ISPs in each city (chosen based on their 1-multihoming performance). This has a negligible impact on our results, as our analysis showed that the performance of the top 20 ISPs is virtually indistinguishable from the performance using all available ISPs in the city (these results are omitted).

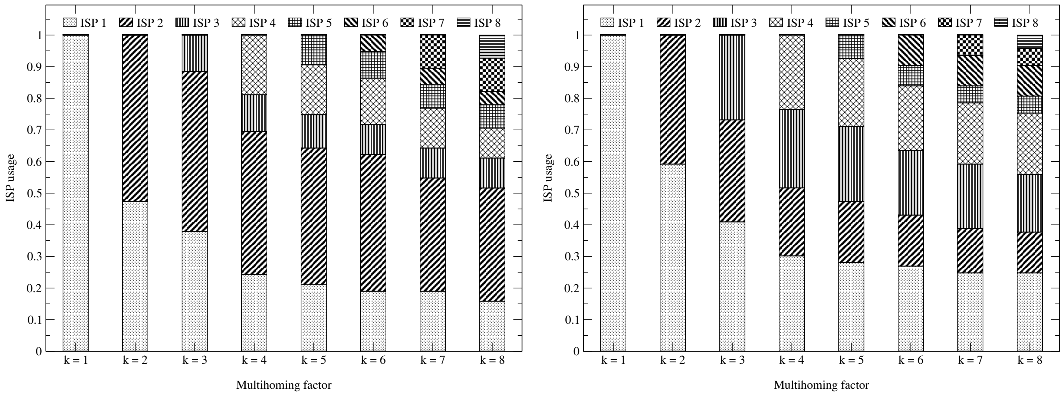
In Figure 4.3(b), we show the potential performance benefits from naive k-multihoming for the first week of data in $\mathcal{H}_1$ (again, the second week's results are similar). Notice that $k > 1$ provides significantly better performance than 1-multihoming in most locations. For a few cities, however, the performance benefit is not as substantial due to a single ISP providing the best performance almost all the time (e.g., Los Angeles). Also, beyond $k = 3$ the benefit from naive k-multihoming is only marginally better than at smaller values of k for most cities.

Table 4.1 shows the order in which ISPs get added to the k-multihoming solution in New York for increasing values of k. For each ISP, we also show its 1-multihoming rank and performance. Notice that the best k-multihoming solution does not necessarily comprise the k best 1-multihoming options (e.g., the third ISP has a rank of 9). Rather, ISPs are added based on their contribution to the overall k-multihoming performance.

We also consider how often each of the ISPs is employed in the optimal schedule for naive multihoming. In particular, we are interested in whether an ISP's contribution toward performance

ISP	Rank	1-multihoming performance	k-multihoming performance
ISP 1	1	1.72	1.72
ISP 2	2	1.93	1.33
ISP 3	9	2.61	1.17
ISP 4	3	2.05	1.09
ISP 5	4	2.29	1.07
ISP 6	19	3.16	1.04
ISP 7	17	3.03	1.03
ISP 8	13	2.93	1.03

Table 4.1: Rank vs overall performance Ranks of the ISPs in the k-multihoming solutions at New York, $k \leq 8$, in the order in which they are added, along with the incremental performance improvement.



(a) Boston (b) New York

Figure 4.4: Relative utilization of ISPs: For the cities of Boston and New York, respectively, the graphs show the fraction of time the ISPs in the naive k-multihoming solutions at the city are utilized in the optimal schedule.

improvement is proportional to the frequency with which it is used in the optimal schedule. The results for two cities, Boston and New York, are illustrated in Figure 4.4. The results show clearly that that the contribution to performance is not proportional to the usage. For example, the 6th ISP in New York is used for a significant fraction of time in the 6-multihoming solution (Figure 4.4(b)). However, the marginal benefit of adding this ISP to the 5-multihoming solution was less than 0.02 (Figure 4.3(b)). It is also possible that an ISP belonging to the best k-multihoming solution is utilized for a very small fraction of time in the optimal schedule, but, whenever used, contributes significantly to improving the overall performance. For example, ISP1 is used for smaller fraction of time than ISP2 for the best naive 2-multihoming solution in Boston (Figure 4.4(a)). However, the contribution of ISP1 to the overall benefit due to 2-multihoming is clearly larger than that of ISP2.

To summarize, even naive multihoming can substantially improve the turnaround time performance of subscriber networks. We also find that the contribution of an ISP to the overall improvement is not proportional to how often it is used in the optimal schedule. Finally, the best set of ISPs does not necessarily include the individual best ISPs. We explore this issue further in Section 4.3.

4.2 2-Multihoming: RTT Performance

In the previous section, we illustrated how naive multihoming could improve the average RTT performance of a subscriber network. While useful to illustrate the RTT benefits of multihoming, this analysis does not highlight the maximum possible performance improvements due multihoming. This arises from a few key limitations of the dataset: the RTT performance data is averaged across many content providers over the duration of an hour; A finer-grained control over the use of ISP links could further improve the RTT performance. Moreover, multihoming could also improve transfer speeds significantly, and provide quick failover. We need new datasets to analyze these benefits of multihoming. In this, and the subsequent sections, we analyze *active* measurement data collected at selected Akamai server and monitoring nodes to address the above drawbacks. First, in this section, we consider a setting in which the end-network has two ISP connections. We illustrate the RTT benefits from intelligently controlling which of the two ISPs is used, at much finer time granularity (e.g., on the order of a few minutes). We refer to this as *2-multihoming*.

4.2.1 Measurement Dataset

To analyze 2-multihoming, we collect RTT performance statistics at 27 geographically distributed Akamai monitoring nodes. One or two monitoring nodes are located in major cities in the U.S., with multiple nodes in the same city attached to different upstream ISPs, as shown in Figure 4.5. Every 6 minutes, these nodes download designated objects directly from a large number of content providers that are Akamai customers. For each attempted download, the performance monitor logs a number of statistics, including the HTTP response code, turnaround time for the request (if successful), the size of the object downloaded, the total response time, and any errors (if unsuccessful).

We focus, in particular, on the *turnaround time*, which is defined as the time between the transfer of the last byte of the request from the Akamai node and the receipt of the first byte of the response from the origin server. Hence, the turnaround time offers a reasonable estimate of network delay. We collected these statistics at all 27 performance monitors for downloads made from about 80 content providers which are customers of Akamai. The data was collected between Thursday, 23^{rd} January, 2003 and Sunday, 26^{th} January, 2003 (inclusive). Of the 80 content providers, 20 are the top customers of Akamai; that is, those for which the Akamai network serves the largest number of bytes.

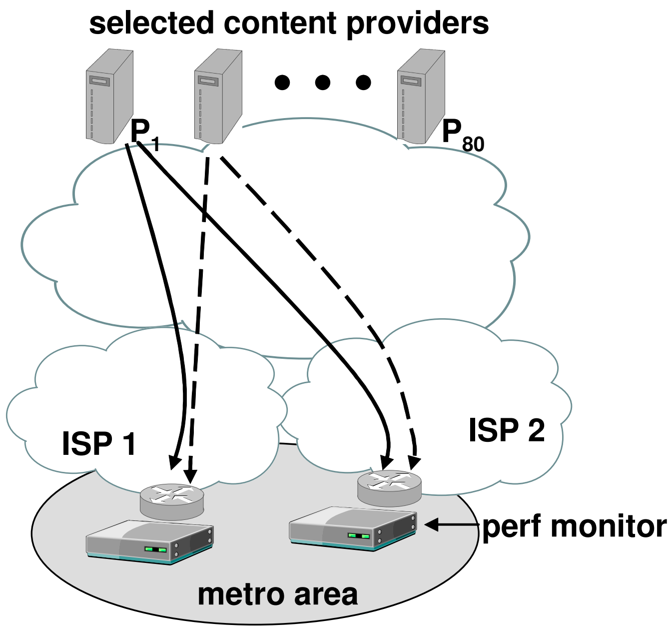


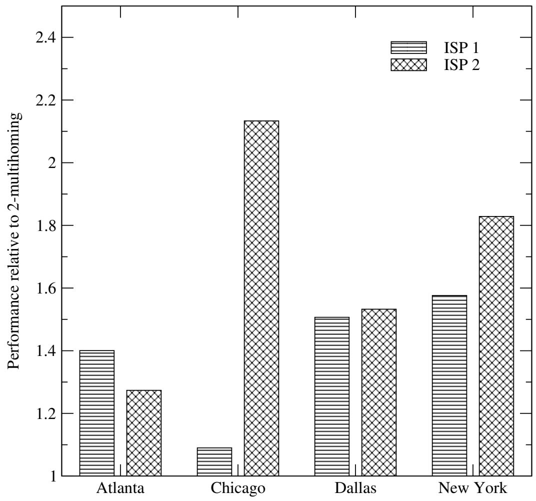
Figure 4.5: 2-Multihoming: Akamai performance monitors in a given city are connected to different ISPs and download 10KB objects at 6-minute intervals from servers belonging to 80 content providers.

4.2.2 Measurement Results: 2-Multihoming

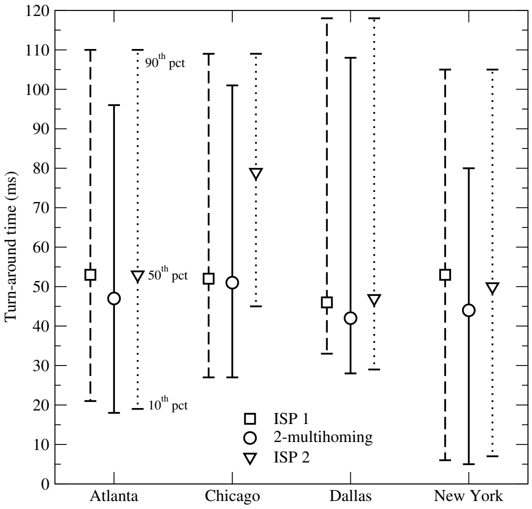
We use the measurements in the above dataset to compare the performance achieved by using the best ISP link for each download, relative to that of using a single ISP for all downloads. We average this ratio over downloads from all of the content providers and report this normalized performance metric. We must also be careful to compare only those transactions for which both performance monitors successfully downloaded the object at roughly the same time. We select cities in the U.S. with 2 performance monitors, giving us four locations: Atlanta, Chicago, Dallas and New York. The rest of the cities have only one performance monitor. The monitor nodes in these cities can then be used to measure the benefits of 2-multihoming employing the respective ISPs. More formally, the computation may be expressed as:

$$N_X \quad = \quad \frac{\sum_{i,t}(M_X(P_i,t)/M_{best}(P_i,t))}{Numvalid(P_i,t)}$$

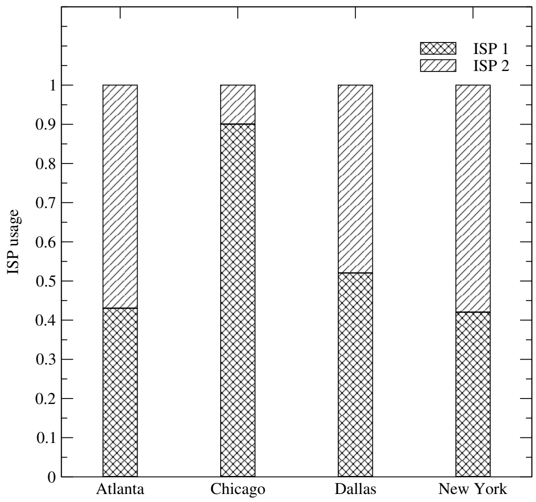
where N_X is the performance of using ISP X, relative to 2-multihoming. $M_X(P_i,t)$ denotes the value of the turnaround time for the transfer initiated at time t by the monitor node attached to ISP X to retrieve an object from content provider P_i. Similarly, $M_{best}(P_i,t)$ is the best value (across both ISPs) of the turnaround time for a transfer to the same city at time t from content provider P_i. The sum in the numerator is over all P_i, t pairs such that there was a transfer logged at time



(a) Potential benefit



(b) Variability of benefits



(c) Relative usage of the ISPs

Figure 4.6: 2-multihoming evaluation: The average benefits are shown in (a). Graph (b) shows the median, 10th and 90th percentile turnaround times for each ISP and for 2-multihoming. The relative usage of the two ISPs in the optimal schedule is shown in (c).

t to content provider P_i via *both* the ISPs A and B. $Numvalid(P_i, t)$ is a function that simply counts the total number of such P_i, t pairs. Notice that the optimal value of N_X is 1 and this occurs whenever one of the two ISPs is consistently better than the other. If $N_X > 1$, then $N_X - 1$ denotes the maximum improvement in performance possible from multihoming to both the ISPs (*i.e.*, from 2-multihoming), compared to the performance seen while using ISP X alone.

The results for the performance benefits from 2-multihoming are shown in Figure 4.6(a), which indicates the value of N_X for the two ISPs. The ISPs in the four cities were tier-1 ISPs. In all four cities, 2-multihoming clearly offers performance benefits, albeit to varying degrees. For example, Chicago's ISP1 provides nearly optimal performance by itself ($N_{ISP1} = 1.09$). However, in the other cities, the minimum performance benefit from 2-multihoming is at least 25% on average.

Figure 4.6(b) illustrates the absolute RTT improvement for the median, 10th, and 90th percentile turnaround time. Note that 2-multihoming uniformly improves the maximum turnaround times, but has less effect on the median and minimum performance. Also, the extent of the improvement varies across cities. Figure 4.6(c) shows the fraction of time when one of the two ISPs provides better performance than the other. Except in Chicago where $ISP1$ is used almost 90% of the time, both the ISPs in the other cities are used for roughly equally in the optimal schedule.

To summarize, using the example of 2-multihoming, we showed that a fine-grained control over ISP connections can result is significant improvements for the subscriber network. Again, the two ISP connections may not be used equally in the optimal schedule for traffic.

4.3 k-Multihoming, $k \geq 2$

The previous two sections provided examples of the benefits of using multiple ISP connections. However, due to limitations of the datasets, a number of more general questions still remain unanswered. For example: (1) How does multihoming improve transfers speeds? (2) Do the performance improvements depend on the destination, time-of-day or the day-of-week? And, (3) How much impact does the exact choice, or number, of the ISPs have on the subscriber performance?

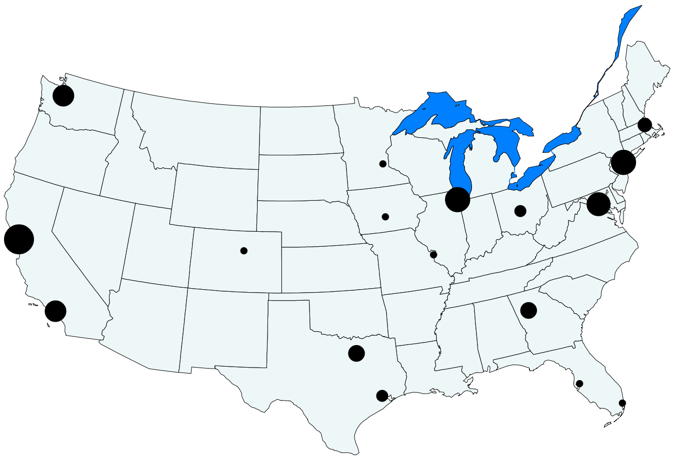
In this section, we address the above questions via comprehensive active measurements of RTT and throughput performance taken over a large testbed consisting of nodes belonging to the server infrastructure of the Akamai CDN. Following a similar methodology to that described in 4.1, we *emulate* a multihoming scenario by selecting a few nodes in a metropolitan area, each singly-homed to a different ISP, and use them collectively as a stand-in for a multihomed network.

The 68 nodes in our testbed span 17 cities in the continental U.S., averaging about four nodes per city, connected to commercial ISPs of various sizes. As before, the nodes are chosen to avoid multiple servers attached to the same ISP in a given city. The list of cities and the tiers of the corresponding ISPs are shown in Figure 4.7(a). The tiers of the ISPs are derived from the work in [108]. The geographic distribution of the testbed nodes is illustrated in Figure 4.7(b). We emulate multihomed networks in 9 of the 17 metropolitan areas where there are at least 3 ISPs – Atlanta, Bay Area, Boston, Chicago, Dallas, Los Angeles, New York, Seattle and Washington D.C.

In what follows, we first describe our data collection methodology. Then, we present the key measurement observations in the following order. First, we present the improvements in RTT and throughput performance from using k-multihoming, $k \geq 2$. Second, we explore whether the improvements due to multihoming are skewed by certain destinations, time of the day or day of the week. Finally, we explore the impact of a suboptimal choice of ISPs on the observed subscriber performance.

City	ISPs/tier				
	1	2	3	4	5
Atlanta, GA	2	0	1	1	0
Bay Area, CA	5	0	3	1	2
Boston, MA	1	0	1	0	1
Chicago, IL	6	1	0	1	0
Columbus, OH	0	1	0	1	0
Dallas, TX	3	0	0	1	0
Denver, CO	1	0	0	0	0
Des Moines, IO	0	1	0	0	0
Houston, TX	1	1	0	0	0
Los Angeles, CA	3	0	3	0	0
Miami, FL	1	0	0	0	0
Minneapolis, MN	0	0	1	0	0
New York, NY	3	2	2	1	0
Seattle, WA	2	0	2	1	1
St Louis, MO	1	0	0	0	0
Tampa, FL	0	1	0	0	0
Washington DC	3	0	3	0	2

(a) Testbed ISPs



(b) Node locations

Figure 4.7: Testbed details: The cities and distribution of ISP tiers in our measurement testbed are listed in (a). The geographic location is shown in (b). The area of each dot is proportional to the number of nodes in the region.

4.3.1 Data Collection

We draw our observations from two datasets collected on the testbed described above. The first data set consists of active HTTP downloads of small objects (10 KB) to measure the turnaround times between the pairs of nodes. Every 6 minutes, we collect turnaround time samples between all pairs of nodes in our testbed (including those within the same city). The second data set contains throughput measurements from active downloads of 1 MB objects between the same set of node-pairs. These downloads occur every 30 minutes between all node-pairs. Throughput is simply the size of the transfer (1 MB) divided by the time between the receipt of the first and last bytes of the response data from the server (source). Notice that this may not reflect the steady-state TCP throughput along the path.

Since our testbed nodes are part of Akamai's production infrastructure, we limit the frequencies at which all-pairs measurements are collected as described above. To ensure that all active probes between pairs of nodes observe similar network conditions, we scheduled them to occur within a 30 second interval for the round-trip time data set, and within a 2 minute interval for the throughput data set. For the latter, we also ensure that an individual node is involved in at most one transfer at any time so that our probes do not contend for bandwidth at the source or destination network. The transfers may interfere elsewhere in the Internet, however. Also, since our testbed nodes are all located in the U.S., the routes we probe, and consequently, our observations, are U.S.-centric.

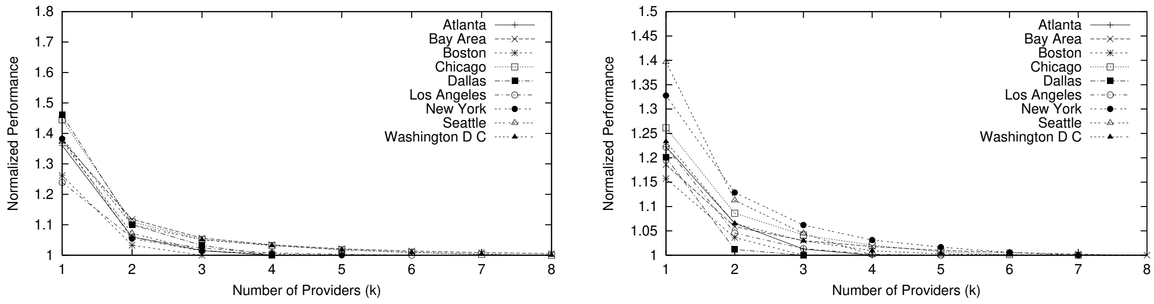
The round-trip time data set was collected from Thursday, December 4th, 2003 through Wednesday, December 10th, 2003. The throughput measurements were collected between Thursday, May 6th, 2004 and Tuesday, May 11th, 2004 (both days inclusive).

4.3.2 k-Multihoming Improvements

To understand performance benefits of k-multihoming, we adopt a similar methodology as the one described in Section 4.2. For each download, we compare the turnaround time achieved by using the best ISP among all those available in the city, with that of using the best ISP in a candidate multihoming option. We average this ratio over transfers to all testbed machines, and report the minimum normalized performance metric. The minimum is taken over all multihoming options. As before, we only compare transactions with simultaneous successful transfers over all ISPs.

Formally, $M_{best}(A_i, t)$ denotes the best value of the turnaround time for a transfer to testbed node A_i ($i = 1, \ldots, 68$) at time t, across all available ISPs in a city. For a k-multihoming option OP_k, let $M_{OP_k}(A_i, t)$ be the best turnaround time across just the ISPs in the set OP_k. We compute the RTT performance benefit from the option OP_k as follows:

$$RTT_{OP_k} = \frac{\sum_{i,t}(M_{OP_k}(A_i, t)/M_{best}(A_i, t))}{Numvalid(t)}$$



(a) Turnaround time

(b) Throughput

Figure 4.8: k-Multihoming Benefits: Figure (a) plots the improvement in web turnaround times from k-multihoming. Figure (b) plots the improvements in throughput.

The sum is over the times t when transfers occur from all the ISPs in the city to node A_i. $Numvalid(t)$ is the number of such time instances. We compute the throughput benefits in a similar fashion:

$$Thru_{OP_k} = \frac{\sum_{i,t}(M_{best}(A_i,t)/M_{OP_k}(A_i,t))}{Numvalid(t)}$$

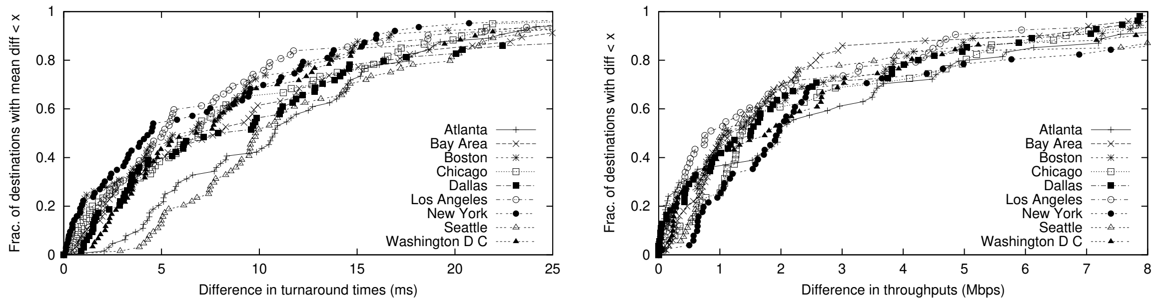
The slight difference in the definition of the RTT and throughput metrics arises from the fact that we are interested in how multihoming can provide *lower* RTTs and *higher* transfers speeds.

In Figure 4.8, we plot the normalized RTT and throughput benefits from k-multihoming as a function of the number of ISPs. Two key facts are apparent from Figure 4.8(a):

- The average RTT improves dramatically when the subscriber uses the best set of 2 or more ISPs, relative to using the single best ISP. The normalized performance metric is lowered by 0.4, reflecting an average 25% improvement in RTTs. Intuitively, this occurs because a second, well-chosen ISP could potentially double the diversity in paths to various destinations. This improved choice in paths helps end networks avoid serious performance problems along any single ISP's paths.

- There is a strong evidence of diminishing returns. Beyond 3 or 4 ISPs, the marginal benefits from using additional ISPs is small. Again, intuitively, this occurs because a fourth or a fifth ISP can provide very little additional diversity than what a well-chosen set of 3 ISPs already provides.

In terms of throughput, multihoming improves performance by as much as 20% relative to a single ISP (see Figure 4.8(b)). Again, we notice diminishing returns in performance beyond 3 ISPs.



(a) Turnaround time (b) Throughput

Figure 4.9: Per-destination performance: Figures (a) and (b) plot the absolute improvements in RTT and throughput performance, respectively, from 3-multihoming relative to 1-multihoming.

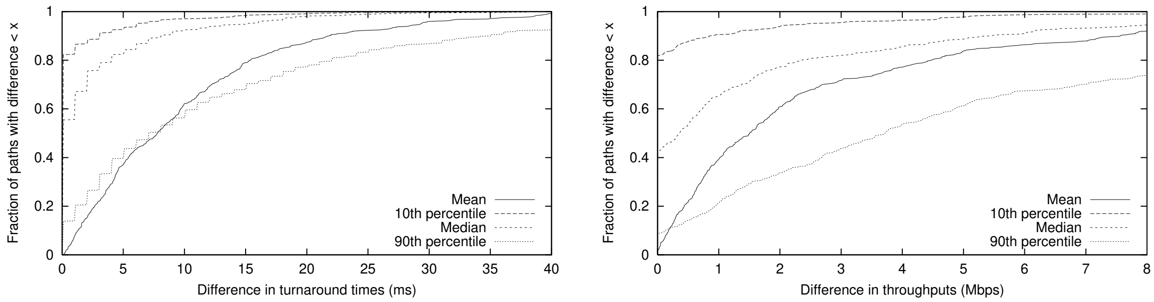
4.3.3 Unrolling the Averages

So far, we presented averages of the performance improvements from multihoming route control. In this section, we present the underlying distributions in the performance improvements. Our goal is to understand if the averages are particularly skewed by: (1) certain destinations, for each source city or (2) a few measurement samples on which multihoming offers significantly better performance than a single ISP or (3) by time-of-day or day-of-week effects.

Performance per destination. In Figure 4.9(a), we show the distribution of the average difference between the best 3-multihoming path and the path via the single best ISP (i.e., each point represents one destination) for various cities. To illustrate, for a subscriber in Seattle, 3-multihoming improves the average RTT per destination by more than 10ms for about 60% of the destinations, and more than 15ms for about 30% of the destinations. In Los Angeles, the improvement due to multihoming is less dramatic. For about 60% of the destinations, the improvement in the average RTT due to multihoming is under 5ms. The key point to notice, however, is that for the 9 cities we consider, there exist a few destinations to which multihoming can offer significantly improved RTT performance.

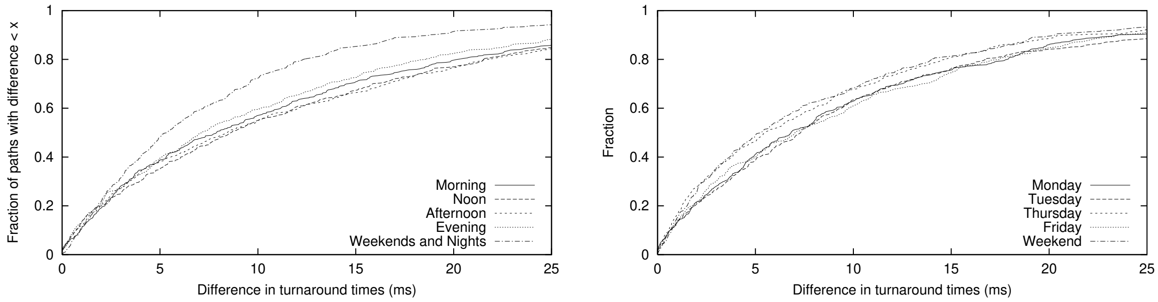
In Figure 4.9(b), we consider the distribution of the average throughput difference of the best 3-multihoming path and the best single ISP. We see that the throughput difference is more than 3 Mbps for 15–40% of the destinations. We also note that, for 1–10% of the destinations, the difference is in excess of 8 Mbps. As with RTT, these observations imply that the transfer speeds to certain destinations could be substantially higher when the subscriber is multihomed.

Mean versus other statistics. In Figures 4.10(a) and (b) we plot the average, median, and 10th and 90th percentiles of the difference in RTT and throughput, respectively, between 3-multihoming and 1-multihoming. In Figure 4.10(a) we see that the median RTT difference is fairly small. More than 90% of the median RTT differences are less than 10ms. However, the 90th percentile of the difference is much higher with roughly 25% greater than 20ms. The 90th percentile throughput differences in Figure 4.10(b) are also significant – more than 8 Mbps about 25% of the time. Con-



(a) Turnaround time (b) Throughput

Figure 4.10: Underlying distributions: Figure showing the mean, median, 10th percentile and 90th percentile difference across various source-destination pairs. Figure (a) plots RTT, while figure (b) plots throughput (pessimistic estimate).

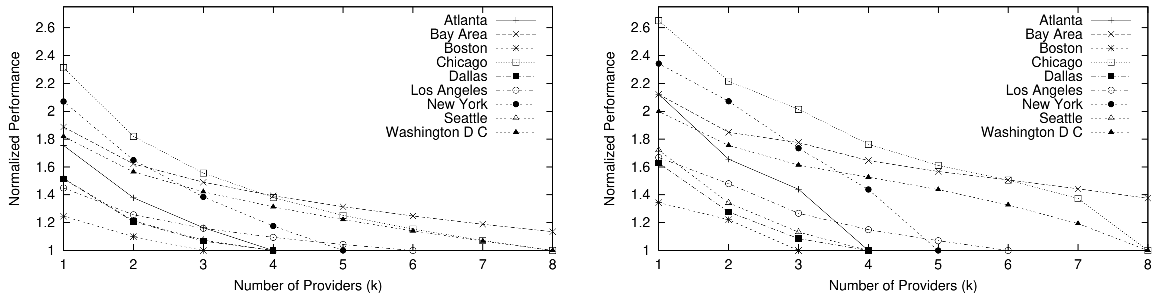


(a) Time-of-Day effects (b) Day-of-Week effects

Figure 4.11: Time of day effects: Figures plotting the impact of the time-of-day (Figure (a)) and day-of-week (pessimistic, Figure (b)) on RTT performance. All times are in EDT.

sidering the median throughput differences, we see that a significant fraction (about 20%) are greater than 3 Mbps. These observations suggest that while multihoming improves the overall performance of all transfers by modest amounts, the performance of a small yet significant fraction could be improved significantly by carefully scheduling traffic across ISP links.

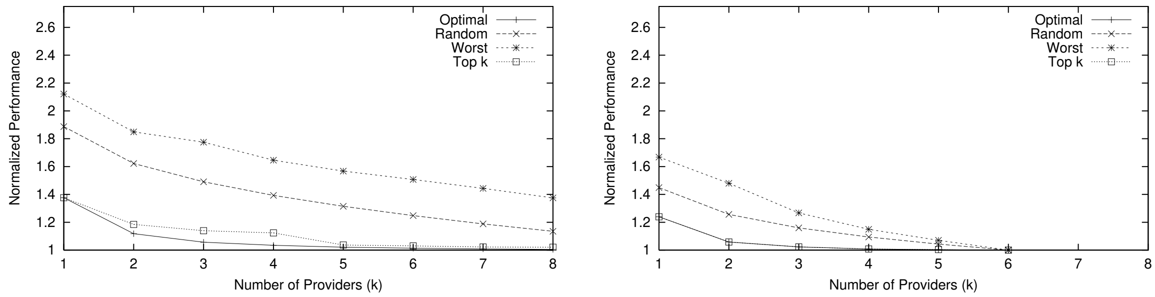
Time-of-day and day-of-week effects. We also consider the effects of hourly and daily network usage patterns on the relative performance of 3-multihoming and 1-multihoming. It might be expected that 1-multihoming would perform particularly worse during peak periods. In Figure 4.11(a) we examine time-of-day effects on the average difference in round-trip times. Notice that the RTT performance improvement does show a correlation with the time of the day. While the improvement due to careful route selection is minimal in the evenings and weekends, the differences are more pronounced during the remaining time-periods. We also examine daily patterns to determine whether the differences are greater during particular days of the week (Figure 4.11(b)). The correlation between the performance improvements and the days of the week is not as significant. As expected,



(a) Performance from a random choice

(b) Performance from a poor choice

Figure 4.12: Impact of sub-optimal choices: Graph (a) shows the expected RTT performance metric from a random k-multihoming option. Graph (b) shows the performance of the worst k-multihoming option.



(a) San Francisco

(b) Los Angeles

Figure 4.13: Choice of ISPs: Figures (a) and (b) show the RTT performance from various ISP selection policies for San Francisco and Los Angeles, respectively.

we observe the improvements to be inferior during weekends. However, the improvements for the other days of the week are not substantially different.

4.3.4 Impact of the Choice of ISPs

In Figure 4.12, we illustrate the impact of choosing a sub-optimal set of ISPs for a k-multihoming solution. We assume that, given a choice of ISPs, a subscriber always uses the best ISP among the available set for its transfers. Comparing Figures 4.12 and (a) 4.8(a), for $k \leq 4$, we see that the RTT performance metric due a random choice of k ISPs is more than 50% higher (e.g., $k = 2$ for Chicago). The difference between optimal and random choices of ISPs is substantial even for higher values of k. In Figure 4.12(b) we show the performance of the worst k-multihoming option. A poor choice of upstream ISPs could result in performance that is at least twice as bad as the optimal choice (compare, e.g., $k = 2$ for Chicago, in Figures 4.12(b) and 4.8(a)). Therefore, while multihoming offers potential for significant performance benefits, it is crucial to carefully choose the right set of ISPs.

Finally, we explore the relative RTT performance from various strategies for selecting ISPs. In particular, Figure 4.13(a) compares the RTT performance metric of optimal, random and worst-case choice of multihoming ISPs for a subscriber in San Francisco. In addition, we also show the RTT performance metric for the case when the subscriber multihomes to the top k individual ISPs (in terms of average RTT performance). Not only does selecting the top k individual ISPs out perform a random choice, it also provides similar RTT performance as the optimal choice. Nevertheless, a more informed selection of ISPs (than simply choosing the top k) could yield up to 5-10% better RTT performance on average (see, for example, $k = 3, 4$ in Figure 4.13(a)). We show a similar set of results for Los Angeles in Figure 4.13(b). In this case, choosing the top k ISPs yields identical RTT performance as the optimal choice of ISPs.

In summary, we find that k-multihoming can substantially improve RTT and throughput performance of subscriber networks. However, there is little additional benefit from employing more than 3 ISP connections. Also, the 3 ISPs themselves must be chosen carefully to realize the potential benefits. In view of this, a good heuristic for the subscriber network is to select the top three individual ISPs serving its city.

4.4 Resilience to Path Failures

End networks employing a single ISP connection must use the paths provided by their ISPs to maintain connectivity with Internet destinations. As a result, failures inside their ISPs' networks inevitably result in connectivity disruptions at the end networks. Past studies (see, for example, [39]) have shown that such failures could last up to several minutes, and severely impact end-user Internet access experience. Such end-networks can vastly improve their resilience from service interruptions by relying on multihoming route control. By monitoring the availability of paths via each of their ISPs and cleverly scheduling traffic on ISPs with available paths, end-network could substantially reduce connectivity outages. In this section, we analyze two distinct datasets collected over the testbed described in Section 4.3 (see Figure 4.7) to quantify the improvements in resilience from using multiple ISP connections. We focus on the special case where the subscriber employs three or fewer ISPs for multihoming.

4.4.1 Active Measurements of Path Availability

In our first approach, we perform two-way ICMP pings between the 68 nodes in our testbed (Figure 4.7). The ping samples were collected between all node-pairs over a five day period from January 23rd, 2004 to January 28th, 2004. The probes are sent once every minute with a one second timeout. If no response is received within a second, the ping is deemed lost. A path is considered to have failed if at least 3 consecutive pings (each one minute apart) from the source to the destination are lost. From these measurements we derive "failure epochs" on each path. The epoch begins when the

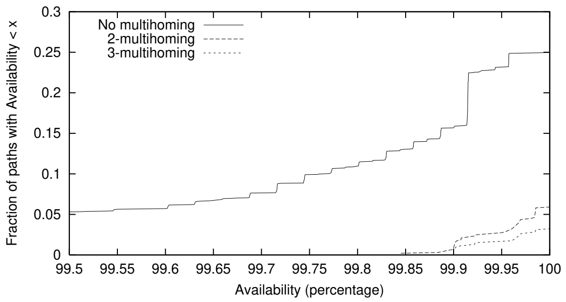


Figure 4.14: End-to-end failures: Distribution of the availability on the end-to-end paths, with and without multihoming. The ISPs in the 2- and 3-multihoming cases are the best 2 and 3 ISPs in each city based on RTT performance, respectively.

third failed probe times out, and ends on the first successful reply from a subsequent probe. These epochs are the periods of time when the route between the source and destination may have failed.

This method of deriving failure epochs has a few limitations. Firstly, since we wait for three consecutive losses, we cannot detect failures that last less than 3 minutes. As a result, our analysis does not characterize the ability of multihoming to avoid such short failures. Secondly, ping packets may also be dropped due to congestion rather than path failure. Unfortunately, from our measurements we cannot easily determine if the losses are due to failures or due to congestion. Finally, the destination may not reply with ICMP echo reply messages within one second, causing us to record a loss. To mitigate this factor, we eliminate paths for which the fraction of lost probes is $> 10\%$ from our analysis. Due to the above reasons, the path failures we identify should be considered an over-estimate of the number of failures lasting three minutes or longer.

From the failure epochs on each end-to-end path, we compute the corresponding *availability*, defined as follows:

$$Availability \quad = \quad 100 \times \left(1 - \frac{\sum_i T_F(i)}{T}\right)$$

where, $T_F(i)$ is the length of failure epoch i along the path, and T is the length of the measurement interval (5 days). The total sum of the failure epochs can be considered the observed "downtime" of the path.

In Figure 4.14, we show a CDF of the availability on the paths we measured, with and without multihoming. When no multihoming is employed, we see that all paths have at least 91% availability (not shown in the figure). Fewer than 5% of all paths have less than 99.5% availability. Route control with multihoming significantly improves the availability on the end-to-end paths, as shown by the 2- and 3-multihoming availability distributions. For both 2- and 3-multihoming, we consider the combinations of ISPs providing the best RTT performance in a city. Even when route control uses only 2 ISPs, less than 1% of the paths originating from the cities we studied have an availability under 99.9%. The minimum availability across all the paths is 99.85%, which is much higher than without multihoming. Also, more than 94% of the paths from the various cities to the respective destinations do not experience any observable failures during the 5 day period (i.e.,availability of 100%). With three ISPs, the availability is improved, though slightly.

4.4.2 Path Availability Analysis

Since the vast majority of paths did not fail even once during our relatively short measurement period, our second approach uses statistics derived from previous long-term measurements to ascertain availability. Feamster et al. collected failure data using active probes between nodes in the RON testbed approximately every 30 seconds for several months [39]. When three consecutive probes on a path were lost, a traceroute was triggered to identify where the failure appeared (i.e., the last router reachable by the traceroute) and how long they lasted. The routers in the traceroute data were also labeled with their corresponding AS number and also classified as border or internal routers. We use a subset of these measurements on paths between non-DSL nodes within the U.S. collected between June 26, 2002 and March 12, 2003 to infer failure rates in our testbed. Though this approach has some drawbacks (which we discuss later), it allows us to obtain a view of longer-term availability benefits of route control that is not otherwise possible from direct measurements on our testbed.

We first estimate the availabilities of different router classes (i.e., the fraction of time they are able to correctly forward packets). We classify routers in the RON testbed traceroutes by their AS tier (using the method in [108]) and their role (border or internal router). Note that the inference of failure location is based on router location, but the actual failure could be at the *link or router* attached to the last responding router.

The availability estimate is computed as follows: If $\sum T_F^C$ is the total time failures attributed to routers of class C were observed, and N_d^C is the total number of routers of class C we observed on each path on day d,[2] then we estimate the availability of a router (or attached link) of class C as:

$$Availability_C \quad = \quad 100 \times \left(1 - \frac{\sum T_F^C}{\sum_d N_d^C \times one\_day}\right)$$

In other words, the fraction of time unavailable is the aggregate failure time attributed to a router of class C divided by the total time we expect to observe a router of class C in any path. Our estimates for various router classes are shown in Table 4.2.

To apply the availability statistics derived from the RON data set, we identified and classified the routers on paths between nodes in our testbed. We performed traceroute measurements approximately every 20 minutes between nodes in our CDN testbed from December 4, 2003 to Dec 11, 2003. For our analysis we used the most often observed path between each pair of nodes; in almost

[2] The dataset only included a single successful traceroute per day. Therefore, we assumed that all active probes took the same route each day.

AS Tier	Location	Availability (%)
1	internal	99.940
1	border	99.985
2	internal	99.995
2	border	99.977
3	internal	99.999
3	border	99.991
4	internal	99.946
4	border	99.994
5	internal	99.902
5	border	99.918

Table 4.2: Availability across router classes: Estimated availability for routers or links classified by AS tier and location. We consider a border router as one with at least one link to another AS.

all cases, this path was used more than 95% of the time. Using the router availabilities estimated from the RON data set, we estimate the availability of routes in our testbed when we use multihoming. When estimating the simultaneous failure probability of multiple paths, it is important to identify which routers are shared among the paths so that failures on those paths are accurately correlated. Because determining router aliases was difficult on some paths in our testbed,[3] we conservatively assumed that the routers at the end of paths toward the same destination were identical if they belonged to the same sequence of ASes. For example, if we had two router-level paths destined for a common node that map to the ASes `A A B B C C` and `D D D B C C`, respectively, we assume the last 3 routers are the same (since `B C C` is common). Even if in reality these routers are different, failures at these routers are still likely to be correlated. The same heuristic was used to identify identical routers on paths originating from the same source node. We assume other failures are independent.

A few aspects of this approach may introduce biases in our analysis. First, the routes on RON testbed paths may not be representative of the routes in our testbed, though we tried to ensure similarity by using only using paths between relatively well-connected RON nodes in the U.S. (a good fraction of RON nodes belong to home DSL users). In addition, we observed that the availabilities across router classes in the RON dataset did not vary substantially across different months, so we do not believe the difference in time frames impacted our results. Second, there may be routers or links in the RON data set that fail frequently and bias the availability of a particular router type. However, since traceroutes are initiated only when a failure is detected, there is no way for us to accurately estimate the overall failure rates of all individual routers. Third, it is questionable whether we should assign failures to the last reachable router in a traceroute; it is possible that the *next* (unknown) or

[3] We found that several ISPs block responses to UDP probe packets used by IP alias resolution tools such as Ally [105]

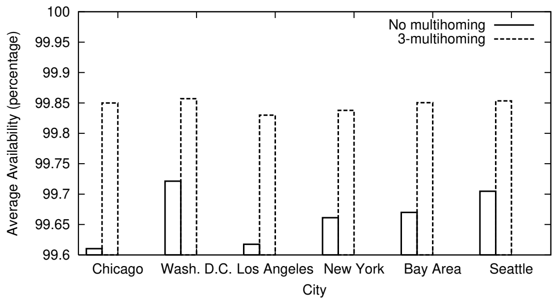


Figure 4.15: Availability comparison: Comparison of availability averaged across paths originating from six cities using a single ISP and using 3-multihoming. ISPs are chosen based on their round-trip time performance.

an even further router in the path is actually the one that failed. Nevertheless, our availabilities still estimate how often failures are observed at or just after a router of a given type.

Figure 4.15 compares the average availability using multihoming route control on paths originating from 6 cities to all destinations in our testbed. As expected from our active measurements, the average availability along the paths in our testbed are relatively high, even for direct paths. 3-multihoming improves the average availability by 0.15-0.24% in all the cities (corresponding to about 13-21 more hours of availability each year). Here, the availability is primarily upper bounded by the availability of the routers or links immediately before the destination that are shared by all three paths as they converge.

In summary, route control can improve the availability of end-networks, in addition to performance. While not all failures can be fully eliminated, the availability offered by route control is, nevertheless, very good for all practical purposes.

4.5 Summary of Observations and their Implications

In this chapter we studied the potential improvements in Internet performance and reliability from multihoming. The key observations from this study are outlined in Table 4.3. Our study establishes that multihoming route control could significantly improve the Internet RTTs, throughputs and reliability of end networks. Furthermore, beyond 3 ISP connections the marginal benefit from employing additional ISPs is minimal. Also, it is important to make an informed choice of ISPs to realize the potential benefits of multihoming. A good strategy is for the end network to multihome to the top 2 or 3 individual ISPs serving its city.

Also, in our evaluation of multihoming route control, we assumed that network destinations were all singly-homed. Despite this restriction, we observed significant performance improvements. In some cases, however, the destination network itself could be multihomed. Furthermore, the destination and the multihomed source network could be administratively related, for example, branch offices of the same parent organization. In such cases, the performance experienced by either end

Multihoming route control can lower Internet RTTs by about 25% or more relative to the best single ISP. In addition, multihoming could yield up to 20% higher transfer speeds.
Our measurements show that substantial fractions of transfers from nine major U.S. cities to various Internet destinations could experience at least an 25ms improvement in RTTs and an 8Mbps improvement in transfer speeds from route control.
In terms of reliability, we observe that multihoming to 2 or 3 ISPs eliminates most failures experienced by an singly-homed network.
We see a strong evidence of diminishing returns. The improvements in Internet performance are marginal beyond 3 ISP connections.
A careful choice of ISPs is important to realize the potential benefits of multihoming route control. A poor, uninformed choice of ISPs, for example, could yield RTTs that are more than double the RTTs from the optimal choice.
In most of the cities we study, employing the top 3 individual ISPs for multihoming provides roughly the same performance improvements as employing the best set of 3 ISPs.

Table 4.3: Benefits of Multihoming Route Control: Summary of key observations regarding multihoming.

when communicating with the order could be heavily optimized by: (1) co-ordinating the choices of the ISPs that the two ends connect to. For example, if both ends connected to the same set of two tier-1 ISP, then all network paths between the two would be at most two AS hops long; And, (2) co-ordinating the choice of routes. For example, if both ends decided to use the same ISP connection for a given data transfer, the transfer would traverse a single ISP domain, avoiding all inter-domain policies and taking the shortest-IP-hop route.

Notice that in Internet routing, each ISP provides a subscriber exactly one BGP path per destination. By employing multihoming, and choosing a good set of ISPs, an end-network enjoys a slightly richer selection of BGP routes per destination (i.e., one route per ISP for every destination), that it can choose from in an informed manner. In effect, therefore, the observations in this chapter have shown that when the routing flexibility of end-networks is improved by moderate amounts using multihoming route control, their Internet performance and reliability can be significantly improved.

The contributions of our measurement study of multihoming can be simply summarized as follows: By improving the routing flexibility of endpoints by moderate amounts, their Internet performance and reliability can be vastly improved. In the next chapter, we ask whether this flexibility is sufficient, or whether enabling much better routing flexibility at end-networks (e.g., using Overlay networks) could result in superior performance and reliability.

Chapter 5

A Comparison of Overlay Routing and Multihoming Route Control

The typical model for enterprise Internet access today is for the enterprise to buy Internet connectivity from an ISP and route its traffic via the ISP. In this model, the ISP determines how the enterprise's data should be routed across the Internet. Often, when the destination involved is connected to a different ISP, the enterprise's own ISP employs Border Gateway Protocol (BGP) to route traffic toward the destination. However, several past analyses of BGP have highlighted serious inefficiencies in its functioning. For example, the routes enabled by BGP often yield sub-optimal RTTs and transfer speeds. Moreover, failures in BGP, resulting from events such as router malfunction, often require unacceptably long reconvergence and stabilization times. Quite aptly, then, these limitations of conventional Internet routing based on the Border Gateway Protocol (BGP) are often held responsible for failures and poor performance of end-to-end Internet transfers.

A number of studies have shown that the underlying connectivity of the Internet is actually capable of providing much greater performance and resilience than endpoints currently receive. These studies (see, for example, Detour [99, 100] and RON [14]) demonstrated that using *overlay routing* to bypass BGP's policy-driven routing enables quicker reaction to failures and improved end-to-end performance. In this approach, endpoints can route their traffic via intermediate overlay nodes deployed around the Internet. This helps endpoints bypass the default routes determined by BGP and avoid performance and availability problems along these routes.

In this chapter, we question whether overlay routing is *required* to make the most of the underlying connectivity, or whether a more intelligent selection of BGP routes at an endpoint is sufficient. Specifically, in this chapter, we compare and contrast overlay routing against multihoming route control. As noted in the previous chapter, multihoming route control enables end-networks to intelligently control and use BGP paths provided by their multiple ISPs. Furthermore, multihoming does not require any changes or improvements to the underlying BGP protocol. Our goal, then, is to answer the following question:

> How much benefit, in terms of Internet performance or resilience, does overlay routing provide over multihoming route control?

If the benefit is small, then BGP path selection (based on multihoming) is not as inferior as it is held to be, and good end-to-end performance and reliability are achievable even when operating completely within standard Internet routing. On the other hand, if overlays yield significantly better performance and reliability characteristics, we can conclude that BGP is fundamentally limited. In such a situation, it is important to develop alternate bypass architectures such as overlay routing or invent new wide-area routing protocols.

Using extensive active downloads and traceroutes between 68 Akamai CDN servers in the testbed described earlier in Section 4.3, we compare multihoming route control and overlay routing in terms of three key metrics: round-trip delay, throughput, and availability. Our measurement results suggest that when route control is employed along with multihoming, it can offer performance similar to overlays in terms of round-trip delay and throughput. On average, the round-trip times achieved by the best BGP paths (selected by an ideal route control mechanism using 3 ISPs) are within 5–15% of the best overlay paths (selected by an ideal overlay routing scheme also multihomed to 3 ISPs). Similarly, the throughput on the best overlay paths is only 1–10% better than the best BGP paths. We also show that the marginal difference in the RTT performance can be attributed mainly to overlay routing's ability to select shorter paths, and that this difference can be reduced further if ISPs implement cooperative peering policies. Our comparison of the end-to-end path availability provided by either approach shows that multihoming route control, like overlay routing, is able to significantly improve the availability of end-to-end paths.

Chapter outline. In Section 5.1 of this chapter, we provide an overview of our approach to comparing overlay routing and route control. In Section 5.2, we analyze the RTT and throughput performance differences between route control and overlay routing and consider some possible reasons for the differences. In Section 5.3, we contrast the end-to-end availability offered by the two schemes. Section 5.4 summarizes the observations in this chapter, discusses the implications of the results, and presents some limitations of our study.

5.1 Terminology

Our objective is to understand whether the modest flexibility of multihoming, coupled with route control, is able to offer end-to-end performance and resilience similar to overlay routing. In order to answer this question, we evaluate an idealized form of multihoming route control driven by the three key assumptions outlined in Chapter 4: perfect information of ISP performance, low-overhead in shifting traffic across ISPs and mechanisms for inbound route control. To ensure a fair comparison, we study a similarly agile form of overlay routing where the endpoint has timely and accurate knowledge of the best performing, or most available, end-to-end overlay paths. Frequent active probing of each overlay link, makes it possible to select and switch to the best overlay path at almost any instant when the size of the overlay network is small ($\sim$50 nodes)[1].

[1] Such frequent probing is infeasible for larger overlays [14].

We compare overlay routing and route control with respect to the degree of flexibility in paths available at the end-network. In general, this flexibility is represented by k, the number of ISPs available to either technique at the end-network. For route control, we consider the notion of k-*multihoming* from Chapter 4, where we evaluate the performance and reliability of end-to-end candidate paths induced by a combination of k ISPs. For overlay routing, we introduce a similar notion of k-*overlays*, where k is the number of ISPs available to an endpoint for any end-to-end overlay path. In other words, this is simply overlay routing in the presence of k ISP connections. When comparing k-multihoming with k-overlays, we report results based on the combination of k ISPs that gives the *best performance* (RTT or throughput) across all destinations.

Figure 5.1 illustrates some possible route control and overlay configurations to clarify the terminology used in this chapter. For example, (a) shows the case of conventional BGP routing with a single default ISP (i.e., 1-multihoming). Figure 5.1(b) depicts endpoint route control with three ISPs (i.e., 3-multihoming). Overlay routing with a single first-hop ISP (i.e., 1-overlay) is shown in Figure 5.1(c), and Figure 5.1(d) shows the case of additional first-hop flexibility in a 3-overlay routing configuration.

When comparing overlay routing and route control, we seek to answer the following key questions:

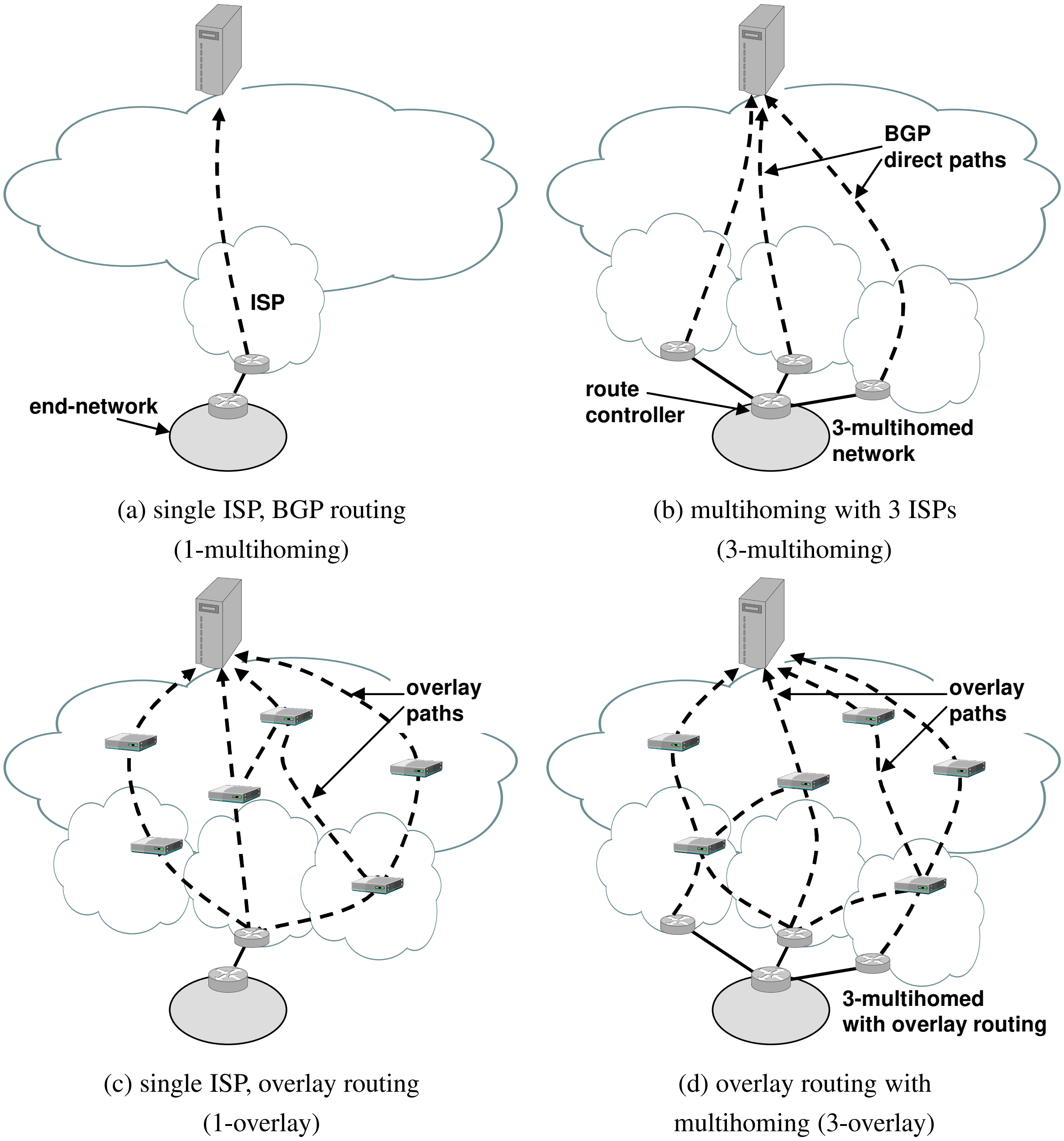
1. On what fraction of end-to-end paths does overlay routing outperform multihoming route control in terms of RTT and throughput? In these cases, what is the extent of the performance difference?

2. What are the reasons for the performance differences? For example, must overlay paths violate inter-domain routing policies to achieve good end-to-end performance?

3. Does route control achieve path availability rates that are comparable with overlay routing?

5.2 Latency and Throughput Performance

We now present our results on the latency and throughput performance benefits of route control compared with overlay routing. We first provide details of our RTT and throughput comparison techniques. Then, we present the key results in the following order. First we compare 1-multihoming against 1-overlays (this is similar to the analysis in [99]). Next, we compare the benefits of using k-overlay routing, relative to using default paths through a single ISP. Then, we compare k-multihoming against 1-overlay routing, for $k \geq 1$. Here, we wish to quantify the benefit to end-systems of greater flexibility in the choice of BGP routes via multihoming, relative to the power of 1-overlays. Next, we contrast k-multihoming against k-overlay routing to understand the additional benefits gained by allowing end-systems almost arbitrary control on end-to-end paths, relative to multihoming. Finally, we examine some of the underlying reasons for the performance differences.



(a) single ISP, BGP routing
(1-multihoming)

(b) multihoming with 3 ISPs
(3-multihoming)

(c) single ISP, overlay routing
(1-overlay)

(d) overlay routing with
multihoming (3-overlay)

Figure 5.1: Routing configurations: Figures (a) and (b) show 1-multihoming and 3-multihoming, respectively. Corresponding overlay configurations are shown in (c) and (d), respectively.

5.2.1 Comparing RTTs and Throughputs

Our comparison of overlays and multihoming is based on observations drawn from the RTT and throughput datasets described earlier in Section 4.3.

RTT Comparison. In the RTT data set, for each 6 minute measurement interval, we build a weighted graph over the 68 testbed nodes where the edge weights are the RTTs measured between the corresponding node-pairs. We then use Floyd's algorithm [31] to compute the shortest paths between all node-pairs. We estimate the RTT performance from using k-multihoming to a given destination by computing the minimum of the RTT estimates along the direct paths from the k ISPs in a city to the destination node (i.e., the RTT measurements between the Akamai CDN nodes representing the k ISPs and the destination node). To estimate the performance of k-overlay routing, we compute the shortest paths from the k ISPs to the destination node and choose the minimum of the RTTs of these paths.

Note that we do not prune the direct overlay edge in the graph before performing the shortest path computation. As a result, the shortest overlay path between two nodes could be a *direct* path (i.e., chosen by BGP). Hence our comparison is not limited to direct versus indirect paths, but is rather between direct and *overlay* paths. In contrast, the comparison in past studies (see, for example, [99]) was between the direct path and the *best indirect path*.

Throughput Comparison. For throughput, we similarly construct a weighted, directed graph between the testbed nodes every 30 minutes (i.e., our 1 MB object download frequency). The edge weights are the throughputs of the 1 MB transfers (where throughput is simply the transfer size divided by the completion time). We compute the throughput performance of k-multihoming and k-overlay routing similar to the RTT performance computation above. Notice, however, that computing the overlay throughput performance is non-trivial and is complicated by the problem of estimating the end-to-end throughput for a 1 MB TCP transfer on indirect overlay paths.

Our approach here is to use round-trip time and throughput measurements on individual overlay hops to first compute the underlying loss rates, using standard model for TCP throughput. Since it is likely that the paths we measure do not observe any loss, thus causing the transfers to likely remain in their slow-start phases, we use the small connection latency model developed in [29][2]. The original model for TCP throughput [83] does not accurately characterize short flows or lossless transfers. Using this model, in our throughput data set, we measure a mean loss rate of 1.2% and median, 90th, 95th and 99th percentile loss rates of 0.004%, 0.5%, 1% and 40% across all paths measured, respectively.

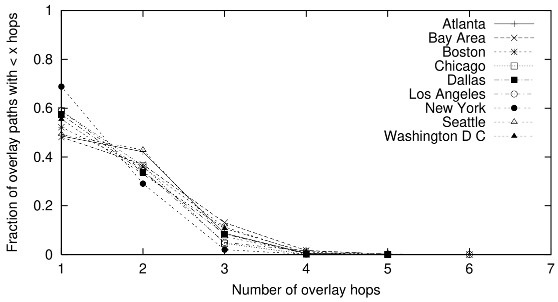
We can then use the sum of round-trip times and a combination of loss rates on the individual hops as the end-to-end round-trip time and loss rate estimates, respectively, and employ the model in [29] to compute the end-to-end overlay throughput for the 1 MB transfers. To combine loss rates on individual links, we follow the same approach as that described in [99]. We consider two possible "combination" or estimation functions:

1. The *optimistic* throughput estimate uses the maximum observed loss on any individual overlay hop along an overlay path as an estimate of the end-to-end overlay loss rate. This assumes

[2]The typical maximum segment size (MSS) in our 1MB transfers is 1460 bytes. Also, the initial congestion window size is 2 segments and there is no initial 200ms delayed ACK timeout on the first transfer.

City	1-multihoming/ 1-overlay
Atlanta	1.35
Bay Area	1.20
Boston	1.28
Chicago	1.29
Dallas	1.32
Los Angeles	1.22
New York	1.29
Seattle	1.34
Wash D.C.	1.30
Average	1.29

(a) 1-multihoming RTT relative to 1-overlays



(b) 1-overlay path length

Figure 5.2: Round-trip time performance: Average RTT performance of 1-multihoming relative to 1-overlay routing is tabulated in (a) for various cities. The graph in (b) shows the distribution of the number of overlay hops in the best 1-overlay paths, which could be the direct path (i.e., 1 overlay hop).

that the TCP sender is primarily responsible for the observed losses.

2. In the *pessimistic* combination, we compute the end-to-end loss rate as the sum of individual overlay hop loss rates, assuming the losses on each link to be due to independent background traffic in the network[3].

Due to the complexity of computing arbitrary length throughput-maximizing overlay paths, we only consider indirect paths comprised of at most two overlay hops in our throughput comparison.

5.2.2 1-Multihoming versus 1-Overlays

First, we compare the performance of overlay routing against default routes via a single ISP (i.e., 1-overlay against 1-multihoming). Our goal is to confirm the observations in past overlay routing studies (such as [99]). Note that, in the case of 1-overlays, the overlay path from a source node may traverse through any intermediate node, including nodes located in the same city as the source.

Round-trip time performance. Figure 5.2(a) shows the RTT performance of 1-multihoming relative to 1-overlay routing. Here, the performance metric (y-axis) reflects the relative RTT from

[3]The end-to-end loss rate over two overlay links with independent loss rates of p_1 and p_2 is $1-(1-p_1)(1-p_2) = p_1 + p_2 - p_1p_2$. p_1p_2 is negligible in our measurements, so we ignore it.

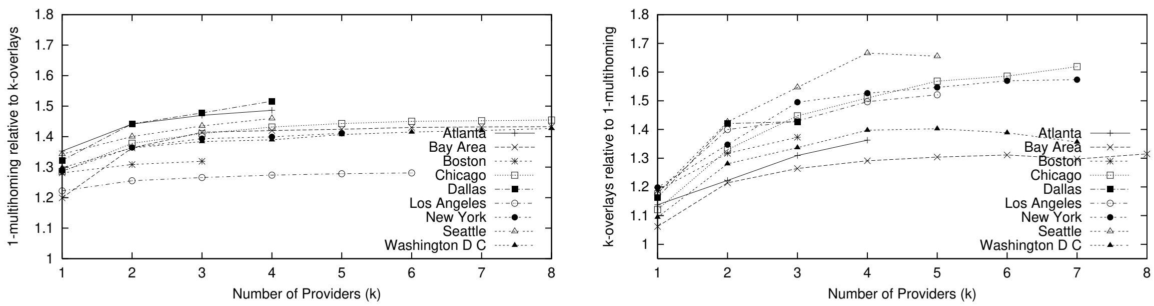
City	Pessimistic estimate		Optimistic estimate	
	Throughput metric	*Fraction of indirect paths*	*Throughput metric*	*Fraction of indirect paths*
Atlanta	1.14	17%	1.17	21%
Bay Area	1.06	11%	1.10	22%
Boston	1.19	22%	1.24	26%
Chicago	1.12	13%	1.15	18%
Dallas	1.16	18%	1.18	22%
Los Angeles	1.18	15%	1.21	17%
New York	1.20	14%	1.25	26%
Seattle	1.18	28%	1.25	35%
Wash D.C.	1.09	13%	1.13	18%
Average	1.15	17%	1.19	23%

Table 5.1: Throughput performance: This table shows the 1 MB TCP transfer performance of 1-overlay routing relative to 1-multihoming (for both estimation functions). Also shown is the fraction of measurements in which 1-overlay routing selects an indirect path in each city.

1-multihoming versus the RTT when using 1-overlays, averaged over all samples to all destinations. The difference between this metric and 1 represents the relative advantage of 1-overlay routing over 1-multihoming. Notice also that since the best overlay path could be the direct BGP path, the performance from overlays is at least as good as that from the direct BGP path. We see from the table that overlay routing can improve RTTs between 20% and 35% compared to using direct BGP routes over a single ISP. The average improvement is about 29%. The observations in [99] are similar.

We show the distribution of overlay path lengths in Figure 5.2(b), where the direct (BGP) path corresponds to a single overlay hop. Notice that in most cities, the best overlay path is only one or two hops in more than 90% of the measurements. That is, the majority of the RTT performance gains in overlay networks are realized without requiring more than a single intermediate hop. Also, on an average, the best path from 1-overlays coincides with the direct BGP path in about 55% of the measurements (average y-axis value at x=1 across all cities).

Throughput performance. In Table 5.1, we show the throughput performance of 1-overlays relative to 1-multihoming for both the pessimistic and the optimistic estimates. 1-overlays achieve 6–20% higher throughput than 1-multihoming, according to the pessimistic estimate. According to the optimistic throughput estimate, 1-overlays achieve 10–25% better throughput. In Table 5.1, we also show the fraction of times an indirect overlay path obtains better throughput than the direct path, for either throughput estimation function. Under the pessimistic throughput estimate, on average, 1-overlay routing benefits from employing an indirect path in about 17% of the cases. Under the optimistic estimate, this fraction is 23%.



(a) Relative RTTs (b) Throughput (pessimistic)

Figure 5.3: Benefits of k-overlays: The RTT of 1-multihoming relative to k-overlays is shown in (a) and throughput (pessimistic estimate) of k-overlays relative to 1-multihoming is shown in (b).

To summarize, 1-overlays offer significantly better round-trip time performance than 1-multihoming. The throughput benefits are lower, but still significant. Also, in a large fraction of the measurements, indirect 1-overlay paths offer better RTT performance than direct 1-multihoming paths.

5.2.3 1-Multihoming versus k-Overlays

In this section we compare the flexibility offered by multihoming route control in combination with overlay routing against using default routes via a single ISP (i.e., k-overlays against 1-multihoming).

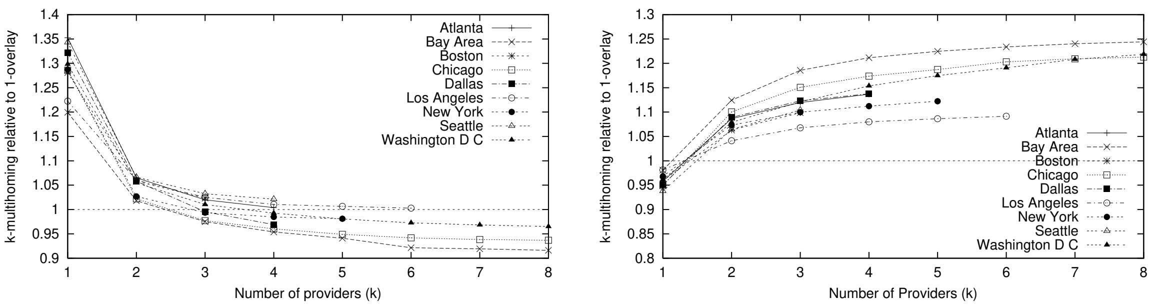
In Figure 5.3(a), we show the RTT performance of 1-multihoming relative to k-overlays as a function of k. Notice that k-overlay routing achieves 25–50% better RTT performance than 1-multihoming, for $k = 3$. Notice also, that the RTT performance from k-overlay routing ($k \geq 3$) is about 5–20% better than that from 1-overlay routing. Figure 5.3(b) similarly compares the throughput performance of k-overlays relative to 1-multihoming, for the pessimistic estimate. Again, 3-overlay routing, for example, is 20–55% better than 1-multihoming and about 10–25% better than 1-overlay routing. The benefit beyond $k = 3$ is marginal across most cities, for both RTT as well as throughput.

In summary, both k-multihoming and k-overlay routing offer much better performance than 1-multihoming, in terms of both RTT and throughput. In addition, k-overlay routing ($k \geq 3$) achieves significantly better performance compared to 1-overlay routing.

5.2.4 k-Multihoming versus 1-Overlays

In the next two sections, we provide a head-to-head comparison of multihoming and overlay routing. First, in this section, we allow endpoints the flexibility of multihoming route control and compare the resulting performance against 1-overlays.

In Figure 5.4, we plot the performance of k-multihoming relative to 1-overlay routing. Here, we compute the average ratio of the best RTT or throughput to a particular destination, as achieved



(a) Relative RTTs (b) Throughput (pessimistic)

Figure 5.4: Multihoming versus 1-overlays: The RTT of k-multihoming relative to 1-overlays is shown in (a) and throughput (pessimistic) of 1-overlays relative to k-multihoming in (b).

by either technique. The average is taken over paths from each city to destinations in other cities, and over time instants for which we have a valid measurement over all ISPs in the city.[4] We also note that in all but three cities, the best 3-multihoming ISPs according to RTT were the same as the best 3 according to throughput; in the three cities where this did not hold, the third and fourth best ISPs were simply switched and the difference in throughput performance between them was less than 3%.

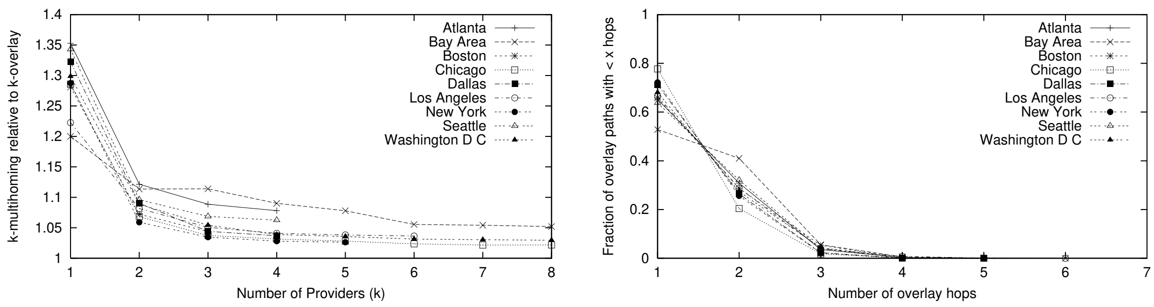
The comparison according to RTT is shown in Figure 5.4(a). The performance advantage of 1-overlays is less than 5% for $k = 3$ in nearly all cities. In fact, in some cities, e.g., Bay Area and Chicago, 3-multihoming is marginally better than overlay routing. As the number of ISPs is increased, multihoming is able to provide shorter round-trip times than overlays. Figure 5.4(b) shows relative benefits according to the pessimistic throughput estimate. Here, multihoming for $k \geq 3$ actually provides 2–12% better throughput than 1-overlays across all cities. The results are similar for the optimistic computation and are omitted for brevity.

In summary, the performance advantages of 1-overlays are vastly reduced (or eliminated) when the endpoint is allowed greater flexibility in the choice of BGP paths via multihoming route control. This implies that multihoming at the first hop is essential to overcome occasional serious problems in the access ISP(s).

5.2.5 k-Multihoming versus k-Overlays

In the previous section, we evaluated 1-overlay routing, where all overlay paths start from a single ISP in the source city. In this section, we allow overlays additional flexibility by permitting them to initially route through more of the available ISPs in each source city. Specifically, we compare the performance benefits of k-multihoming against k-overlay routing.

[4]Across all cities, an average of 10% of the time instants did not have a valid measurement across all ISPs; nearly all of these cases were due to limitations in our data collection infrastructure, and not failed download attempts.



(a) Relative RTTs

(b) 3-Overlay path length

Figure 5.5: Round-trip time improvement: Round-trip time from k-multihoming relative to k-overlay routing, as a function of k, is shown in (a). In (b), we show the distribution of the number of overlay hops in the best k-overlay paths, for k=3.

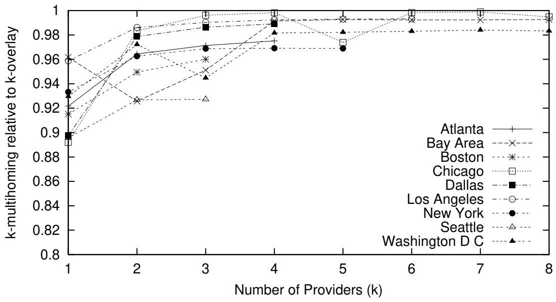
In the case of k-overlays, the overlay path originating from a source node may traverse any intermediate nodes, including those located in the same city as the source. Notice that the performance from k-overlays is at least as good as that from k-multihoming (since we allow overlays to take a direct BGP path. The question, then, is how much more advantage do overlays provide if multihoming is already employed by the source.

Round-trip time performance. Figure 5.5(a) shows the improvement in RTT for k-multihoming relative to k-overlays, for various values of k. We see that on average, for $k = 3$, overlays provide 3–12% better RTT performance than the best multihoming solution in most of the cities in our study. The performance gap between multihoming and overlays is less significant for $k \geq 4$.

Figure 5.5(b) shows the distribution of the number of overlay hops in the paths selected by 3-overlay routing optimized for RTT. The best overlay path coincides with the best 3-multihoming BGP path in 67% of the cases, on average across all cities (the average y-axis value at $x = 1$). Recall that the corresponding fraction for 1-overlay routing in Figure 5.2(b) was 55%. With more ISPs to links to choose from, overlay routing selects a *higher* fraction of direct BGP paths, as opposed to choosing from the greater number of indirect paths also afforded by multihoming.

Throughput performance. Figure 5.6(a) shows the throughput performance of k-multihoming relative to k-overlays using the pessimistic throughput estimation function. From this figure, we see that multihoming achieves throughput performance within 1–10% of overlays, for $k = 3$. The performance improves up to $k = 3$ or $k = 4$. In all the cities, the throughput performance of 4-multihoming is within 3% of overlay routing. In Figure 5.6(b), we also show the fraction of measurements where an indirect 3-overlay path offers better performance than the direct 3-multihoming path, for the pessimistic throughput estimate. On average, this fraction is about 8%. Notice that this is again lower than the corresponding percentage for 1-overlays from Table 5.1 ($\approx 17\%$).

To summarize, when employed in conjunction with multihoming, overlay routing offers marginal benefits over employing multihoming alone. In addition, k-overlay routing selects a larger fraction



City	Fraction of indirect paths
Atlanta	5%
Bay Area	1%
Boston	13%
Chicago	3%
Dallas	8%
Los Angeles	4%
New York	8%
Seattle	31%
Wash D.C.	2%
Average	8%

(a) Throughput improvement (pessimistic estimate)

(b) Fraction of indirect paths in 3-overlay routing

Figure 5.6: Throughput improvement: Throughput performance of k-multihoming relative to k-overlays for various cities is shown in (a). The table in (b) shows the fraction of measurements on which k-overlay routing selected an indirect end-to-end path, for the case of $k = 3$.

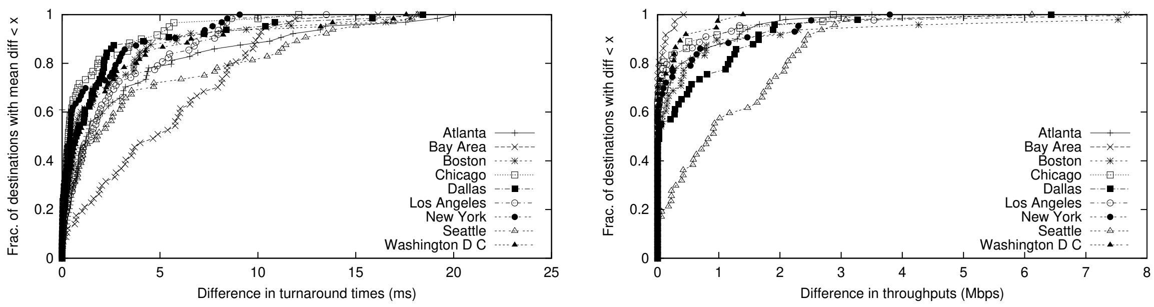
of direct BGP-based end-to-end paths, compared to 1-overlay routing.

5.2.6 Unrolling the Averages

So far, we presented averages of the performance differences for various forms of overlay routing and multihoming route control. In this section, focusing on 3-overlays and 3-multihoming, we present the underlying distributions in the performance differences along the paths we measure. Our goal in this section is to understand if the averages are particularly skewed by: (1) certain destinations, for each source city or (2) a few measurement samples on which overlays offer significantly better performance than multihoming or (3) by time-of-day or day-of-week effects.

Performance per destination. In Figure 5.7(a), we show the distribution of the average difference 3-multihoming path and the best 3-overlay path to destination nodes in the testbed from various origin cities (i.e., each point represents one destination). In most cities, the average RTT differences across 80% of the destinations are less than 10ms. Notice that in most cities, the difference is greater than 15ms for less than 5% of the destinations.

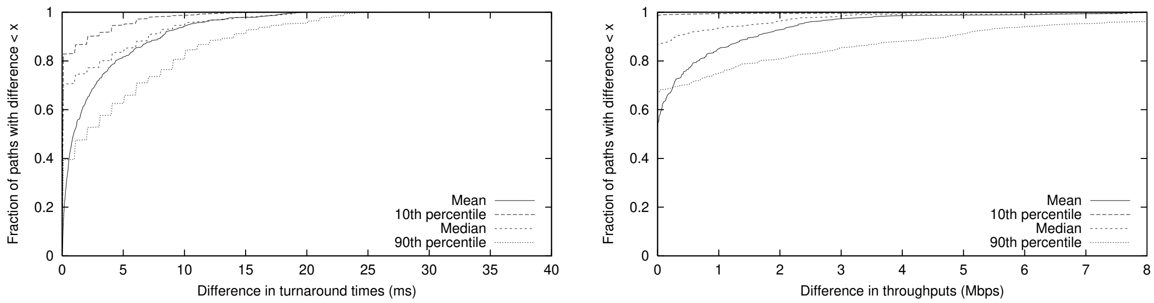
In Figure 5.7(b), we consider the distribution of the average throughput difference of the best 3-multihoming path and the best 3-overlay path for the pessimistic estimate of throughput. We see the throughput difference is less than 1 Mbps for 60–99% of the destinations. We also note that, for



(a) Mean difference in round-trip times

(b) Mean difference in throughputs (pessimistic)

Figure 5.7: Performance per destination: Figure (a) is a CDF of the mean difference in RTTs along the best overlay path and the best direct path, across paths measured from each city. Similarly, Figure (b) plots the CDF of the mean difference in throughputs (pessimistic estimate).



(a) Round-trip time

(b) Throughput (pessimistic)

Figure 5.8: Underlying distributions: Figure showing the mean, median, 10th percentile and 90th percentile difference across various source-destination pairs. Figure (a) plots RTT, while figure (b) plots throughput (pessimistic estimate).

1–5% of the destinations, the difference is in excess of 4 Mbps. Recall from Figure 5.6, however, that these differences result in an average relative performance advantage for overlays of less than 1–10% (for $k = 3$).

Mean versus other statistics. In Figures 5.8(a) and (b) we plot the average, median, and 10th and 90th percentiles of the difference in RTT and (pessimistic) throughput, respectively, between the best 3-multihoming option and the best 3-overlay paths for all cities. In Figure 5.8(a) we see that the median RTT difference is fairly small. More than 90% of the median RTT differences are less than 10ms. The 90th percentile of the difference is marginally higher with roughly 10% greater than 15ms. The median throughput differences in Figure 5.8(b) are also relatively small – less than 500Kbps about 90% of the time. Considering the upper range of the throughput difference (i.e., the 90th percentile difference), we see that a significant fraction (about 20%) are greater than 2 Mbps.

These results suggest that the absolute round-trip and throughput differences between multihoming and overlay routing are small for the most part, though there are a small fraction of cases where differences are more significant, particularly for throughput.

Time-of-day and day-of-week effects. We also consider the effects of daily and weekly network usage patterns on the relative performance of k-multihoming and k-overlays. It might be expected that route control would perform worse during peak periods since overlay paths have greater freedom to avoid congested parts of the network. We do not see any discernible time-of-day effects in paths originating from a specific city, however, both in terms of RTT and throughput performance.

Similarly, we also examine weekly patterns to determine whether the differences are greater during particular days of the week, but again there are no significant differences for either RTT or throughput. We omit both these results for brevity. The lack of a time-of-day effect on the relative performance may indicate that ISP network operators already take such patterns into account when performing traffic engineering.

In summary, k-overlays offer significantly better performance relative to k-multihoming for a small fraction of transfers from a given city. We observed little dependence on the time of the day or day of the week in the performance gap between overlays and multihoming.

5.2.7 Reasons for Performance Differences

Next, we try to identify the causes of performance differences between k-multihoming and k-overlay routing. We focus on the RTT performance, and the case $k = 3$. First, we ask if indirect paths primarily improve propagation delay or mostly select less congested routes than the direct paths. Then, we focus on how often the best-performing indirect paths violate common inter-domain and peering policies.

5.2.7.1 Propagation Delay and Congestion Improvement

In this section, we want to understand whether the modest advantage we observe for overlay routing is due primarily to its ability to find "shorter" (i.e., lower propagation delay) paths outside of BGP policy routing, or whether the gains come from being able to avoid congestion in the network.

The pairwise instantaneous RTT measurements we collect may include a queuing delay component in addition to the base propagation delay. When performance improvements are due primarily to routing around congestion, we expect the difference in propagation delay between the indirect and direct path to be small. Similarly, when the propagation difference is large, we can attribute the performance gain to the better efficiency of overlay routing compared to BGP in choosing "shorter" end-to-end paths. In our measurements, to estimate the propagation delay on each path, we take the 5th percentile of the RTT samples for the path.

In Figure 5.9, we show a scatter plot of the overall RTT improvement (x-axis) and the corresponding propagation time difference (y-axis) offered by the best overlay path relative to the best multihoming path. The graph only shows measurements in which the indirect overlay paths offer

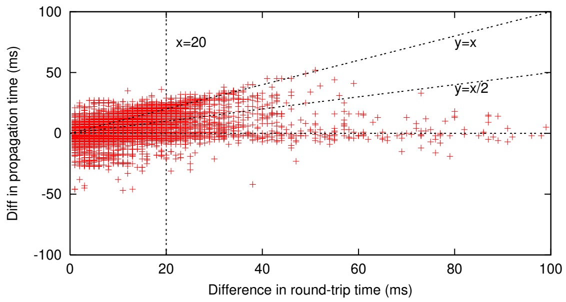


Figure 5.9: Propagation vs congestion: A scatter plot of the RTT improvement (x-axis) vs propagation time improvement (y-axis) of the indirect overlay paths over the direct paths.

an improved RTT over the best direct path. Points near the $y = 0$ line represent cases in which the RTT improvement has very little associated difference in propagation delay. Points near the $y = x$ line are paths in which the RTT improvement is primarily due to better propagation time.

For paths with a large RTT improvement (e.g., > 50ms), the points are clustered closer to the $y = 0$ line, suggesting that large improvements are due primarily to routing around congestion. We also found, however, that 66% of all the points lie above the $y = x/2$ line. These are closer to the $y = x$ line than $y = 0$, indicating that a majority of the round-trip improvements do arise from a reduction in propagation delay. In contrast, Savage et al. [99] observe that both avoiding congestion and the ability to find shorter paths are equally responsible for the overall improvements from overlay routing. The difference in our observations from those in [99] could be due to the fact that Internet paths are better provisioned and less congested today than 3-4 years ago. However, they are sometimes circuitous, contributing to inflation in end-to-end paths (see [104, 110] for detailed accounts of path inflation).

To further investigate the relative contributions of propagation delay and congestion improvements, we focus more closely on cases where indirect overlay paths offer a significant improvement (> 20ms) over the best direct paths. Visually, these are points lying to the right of the $x = 20$ line in Figure 5.9. In Table 5.2 we present a classification of the indirect overlay paths offering > 20ms RTT improvement. Recall that, in our measurement, 33% of the indirect 3-overlay paths had a lower RTT than the corresponding best direct path (Section 5.2.5, Figure 5.5 (b)). However, only 4.8% of these paths improved the delay by more than 20ms (Table 5.2, row 3). For less than half of these, or 2.2% of all lower delay overlay paths, the propagation delay improvement relative to direct paths was less than 50% of the overall RTT improvement. Visually, these points lie to the right of $x = 20$ and below the $y = x/2$ lines in Figure 5.9. Therefore, these are paths where the significant improvement in performance comes mainly from the ability of the overlay to avoid congested links. Also, when viewed in terms of all overlay paths (see Table 5.2, column 3), we see that these paths form a very small fraction of all overlay paths ($\approx 0.7\%$).

Finally, if we consider the propagation delay of the best indirect overlay path versus the best

Total fraction of lower delay overlay paths	33%	
	Fraction of lower delay paths	Fraction of all overlay paths
Indirect paths with $>$ 20ms improvement	4.8%	1.6%
Prop delay improvement $<$ x% of overall improvement (whenever overall improvement $>$ 20ms)		
$<$ 50%	2.2%	0.7%
$<$ 25%	1.7%	0.6%
$<$ 10%	1.3%	0.4%

Table 5.2: Analysis of overlay paths: Classification of indirect paths offering $>$ 20ms improvement in RTT performance.

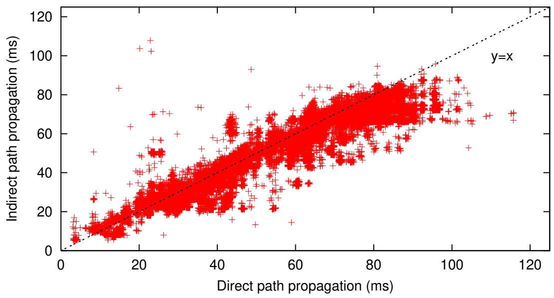


Figure 5.10: "Circuitousness" of routes: Figure plotting the propagation delay of the best indirect path (y-axis) against the best multihoming path (x-axis).

multihoming path, we can get some idea of the ability of either system to avoid overly "circuitous" paths, arising from policy routing, for example. Figure 5.10 shows a scatter plot of the propagation delay of the best direct path from a city (x-axis) and the best propagation delay via an indirect path (y-axis). Again, points below the $y = x$ line are cases in which overlay routing finds shorter paths than conventional BGP routing, and vice versa. Consistent with the earlier results, we see that the majority of points lie below the $y = x$ line where overlays find lower propagation delay paths. Moreover, for cases in which the direct path is shorter (above the $y = x$ line), the difference is generally small, under 10 or 15ms.

In summary, a vast majority of RTT performance improvements from overlay routing arise from its ability to find shorter end-to-end paths compared to the best direct BGP paths. However, the most significant improvements stem from the ability of overlay routing to avoid congested ISP links[5].

5.2.7.2 Inter-domain and Peering Policy Compliance

To further understand the performance gap between some overlay routes and direct BGP routes, we categorize the overlay routes by their compliance with common inter-domain and peering policies. Inter-domain and peering policies typically represent business arrangements between ISPs [42, 80]. Because end-to-end overlay paths need not adhere to such policies, we try to quantify the performance gain that can be attributed to ignoring them.

ISPs typically obey two key inter-domain policies [43]:

1. **Valley-free routing:** ISPs generally do not provide transit between their providers or peers because it represents a cost to them.

2. **Prefer customer routing:** When possible, it is economically preferable for an ISP to route traffic via customers rather than providers or peers, and peers rather than providers.

In addition, Spring et al. [104] observed that ISPs often obey two common *peering policies*:

1. Early exit: ISPs "offload" traffic to peers quickly by using the peering point closest to the source.

2. Late exit: Some ISPs cooperatively carry traffic further than they have to by using peering points closer to the destination.

Also, BGP path selection is impacted by the fact that the routes must have the shortest AS hop count. In this section, we focus on indirect overlay paths (i.e., > 1 virtual hop) that provide better end-to-end round-trip time performance than the corresponding direct BGP paths. To characterize these routes, we identified AS level paths using traceroutes performed over our testbed during the same period as the RTT measurements. Each turnaround time measurement was matched with a traceroute that occurred within 20 minutes of it (2.7% did not have corresponding traceroutes and were ignored in this analysis). We map IP addresses in the traceroute data to AS numbers using a commercial tool which uses BGP tables from multiple vantage points to extract the "origin AS" for each IP prefix [3].

One issue with deriving the AS path from traceroutes is that these router-level AS paths may be different than the actual BGP AS path [69, 13, 51], often due to the appearance of an extra AS number corresponding to an Internet exchange point or a sibling AS[6]. In our analysis, we omit

[5] The improvements from overlay routing could also be from overlays choosing higher bandwidth paths. This aspect is difficult to quantify and we leave it as an open problem.

[6] Two ASes identified as peers may actually be siblings [108, 42], in which case they would provide transit for each other's traffic because they are administered by the same entity. We classified peers as siblings if they appeared to provide transit in the direct BGP paths in our traceroutes, and also manually adjusted pairings that were not related.

	Improved Overlay Paths			>20ms Imprv Paths		
	%	*RTT Imprv (ms)*		%	*RTT Imprv (ms)*	
		Avg	90th		Avg	90th
Violates Inter-Domain Policy	**66.8**	8.3	17	**68.7**	33.7	40
Valley-Free Routing	61.0	8.2	17	58.5	33.7	40
Prefer Customer	14.9	8.9	18	16.3	41.3	47
Valid Inter-Domain Path	**25.2**	7.3	15	**19.4**	36.1	44
Same AS-Level Path	15.3	6.9	13	9.4	40.9	53
Earlier AS Exit	1.9	5.6	10	0.8	43.2	51
Similar AS Exits	6.9	6.4	12	4.9	39.6	55
Later AS Exit	6.5	7.9	14	3.7	42.1	51
Diff AS-Level Path	9.9	8.0	17	10.0	31.5	39
Longer than BGP Path	4.5	7.6	17	4.6	30.9	43
Same Len as BGP Path	4.8	8.6	18	5.3	32.0	37
Shorter than BGP Path	0.6	6.2	9	0.1	36.4	55
Unknown	**8.0**			**11.9**		

Table 5.3: Overlay routing policy compliance: Breakdown of the mean and 90th percentile round trip time improvement of indirect overlay routes by: (1) routes did not conform to common inter-domain policies, and (2) routes that were valid inter-domain paths but either exited ASes at different points than the direct BGP route or were different than the BGP route.

exchange point ASes, and also combine the sibling ASes, for those that we are able to identify. To ascertain the policy compliance of the indirect overlay paths, we used AS relationships generated by the authors of [108] during the same period as our measurements.

In our AS-level overlay path construction, we ignore the ASes of intermediate overlay nodes if they were used merely as non-transit hops to connect overlay path segments. For example, consider the overlay path between a source in AS `S1` and a destination in `D2`, composed of the two AS-level segments `S1 A1 B1 C1` and `C1 B2 D2`, where the intermediate node is located in `C1`. If the time spent in `C1` is short ($<$ 3ms), and `B1` and `B2` are the same ISP, we consider the AS path as `S1 A1 B1 D2`, otherwise we consider it as `S1 A1 B1 C1 B2 D2`. Since we do this only for intermediate ASes that are not a significant factor in the end-to-end round-trip difference, we avoid penalizing overlay paths for policy violations that are just artifacts of where the intermediate hop belongs in the AS hierarchy.

Table 5.3 classifies the indirect overlay paths by policy conformance. As expected, the majority of indirect paths (67%) violated either the valley-free routing or prefer customer policies. However, a large fraction of overlay paths (25%) appeared to be policy compliant. We sub-categorize the latter fraction of paths further by examining which AS-level overlay paths were identical to the AS-level direct BGP path and which ones were different.

For each overlay path that was identical, we characterize it as exiting an AS earlier than the direct path if it remained in the AS for at least 20ms less than it did in the direct path. We characterized it as exiting later if it remained in an AS for at least 20ms longer. We consider the rest of the indirect paths to be "similar" to the direct BGP paths. We see that almost all identical AS-level overlay paths either exited later or were similar to the direct BGP path. This suggests that cooperation among ISPs, e.g., in terms of late exit policies, can improve performance on BGP routes and further close the gap between multihoming and overlays. We also note that for the AS-level overlay paths that differed, the majority were the same length as the corresponding direct path chosen by BGP.

To summarize, in achieving better RTT performance than direct BGP paths, most indirect overlay paths violate common inter-domain routing policies. We observe that a fraction of the policy-compliant overlay paths could be realized by BGP if ISPs employed cooperative peering policies such as late exit.

5.3 Resilience to Path Failures

BGP's policy-based routing architecture masks a great deal of topology and path availability information from end-networks in order to respect commercial relationships and limit the impact of local changes on neighboring downstream ASes. This design, while having advantages, can adversely affect the ability of end-networks to react quickly to service interruptions since notifications via BGP's standard mechanisms can be delayed by tens of minutes [62]. Networks employing multihoming route control can mitigate this problem by monitoring paths across ISP links, and switching to an alternate ISP when failures occur. Overlay networks provide the ability to quickly detect and route around failures by frequently probing the paths between all overlay nodes.

In this section, we perform two separate preliminary analyses of the resilience datasets described earlier in Section 4.4 to assess the ability of both mechanisms to withstand end-to-end path failures and improve availability of Internet paths. As described in Chapter 4, the first approach evaluates the availability provided by route control based on active probe measurements on our testbed. In the second we compute the end-to-end path availability from both route control and overlays using estimated availabilities of routers along the paths.

Our aim is to answer the following questions: Is multihoming route control similar to overlay routing in improving end-to-end fault tolerance? If not, is the resilience from multihoming still reasonable enough to mask almost all failures experienced by the end network?

5.3.1 Active Measurements of Path Availability

We evaluated the ability of multihoming to improve path availability earlier in Section 4.4.1. As noted then, even using two ISPs can significantly improve the availability of Internet paths. For

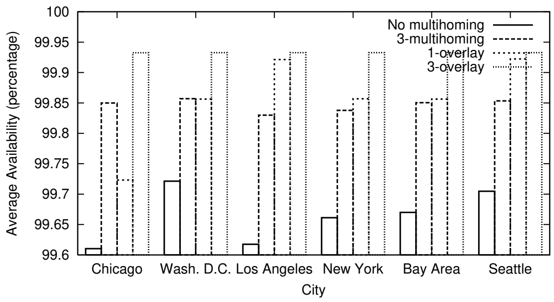


Figure 5.11: Availability comparison: Comparison of availability averaged across paths originating from six cities using a single ISP, 3-multihoming, 1-overlays, and 3-overlays. ISPs are chosen based on their round-trip time performance.

example, less than 1% of the paths originating from the cities we studied have an availability under 99.9% when 2-multihoming was employed. The minimum availability across all the paths is 99.85%, which is much higher than without multihoming (91%). From our analysis of overlay routing, we find that it is indeed able to circumvent even the few failures that route control could not avoid (results omitted for brevity). However, this results in only a marginal improvement over route control which already offers very good availability.

5.3.2 Path Availability Analysis

In Figure 5.11 we compare the average availability using overlays and route control on paths originating from 6 cities to all destinations in our testbed, based on the router failure data presented earlier in Section 4.4.2. For overlay routing, we only calculate the availability of the paths for the first and last overlay hop (since these will be the same no matter which intermediate hops are used), and assume that there is always an available path between other intermediate hops. An ideal overlay has a practically unlimited number of path choices, and can avoid a large number of failures in the middle of the network.

As described earlier, the availability of the paths in our testbed is relatively high overall. 3-multihoming improves the average availability by 0.15-0.24% in all the cities. In most cases, 1-overlays have slightly higher availability (at most about 0.07%). Since a 1-overlay has arbitrary flexibility in choosing intermediate hops, only about 2.7 routers are common (on average) between all possible overlay paths, compared to about 4.2 in the 3-multihoming case. However, note that a 1-overlay path using a single ISP is more vulnerable to access link failures than when multihoming is employed. For example, the low availability of the 1-overlay in Chicago is due to: (1) the chosen ISP (based on RTT performance) is a tier 4 network, which has internal routers with relatively lower availability, and (2) all paths exiting that ISP have the first 5 hops in common and hence have a high chance of correlated failures. Finally, we see that using a 3-overlay usually makes routes only slightly more available than when using a 1-overlay (between 0.01% to 0.08%, excluding Chicago).

This is because at least one router is shared by all paths approaching a destination, so failures at that router impact all possible overlay paths.

In summary, we note that despite the greater flexibility of overlays, route control with 3-multihoming is still able to achieve an estimated availability within 0.08-0.10% (or about 7 to 9 hours each year) of 3-overlay.

5.4 Measurement Caveats, Summary of Observations and their Implications

In this section, we summarize the observations made from our measurements. We highlight other fundamental trade-offs between overlay routing and multihoming route control that are difficult to assess. We also comment on the limitations of our study.

Our key observations as summarized in Table 5.4. As expected, our results show that overlay routing does provide improved latency, throughput, and reliability over multihoming route control. However, the difference between the two systems is *insignificant* for all practical purposes. This is despite the fact that multihoming offers much lesser flexibility in routing than overlays. Moreover, multihoming does not require a third-party deployment and is a purely endpoint based mechanism.

We found that overlay routing's performance gains arise primarily from the ability to find routes that are physically shorter (i.e. shorter propagation delay). In addition, its reliability advantages stem from having at its disposal a superset of the routes available to standard routing. The surprise in our results is that, while past studies of overlay routing have shown this advantage to be large, we found that careful use of a few additional routes via multihoming at the end-network was enough to significantly reduce the advantage of overlays. Since their performance is similar, the question remains whether overlays or multihoming is the better choice. To answer this, we must look at other factors such as cost, deployment issues and future upgrades to BGP routing. We discuss these issues next. In addition, we also outline key constraints imposed by our comparison study.

Cost of operation. Unfortunately, it was difficult to consider the cost of implementing route control or overlays in our evaluation. In the case of multihoming, a stub network must pay for connectivity to a set of different ISPs. We note that different ISPs charge different amounts and therefore the solution we consider "best" may not be the most cost-effective choice. In the case of overlays, we envision that there will be overlay service offerings, similar to Akamai's SureRoute [4]. Users of overlays with multiple first hop choices (k-overlay routing in our analysis) must add the cost of subscribing to the overlay service to the base cost of ISP multihoming.[7] Using an overlay with a single ISP (i.e., 1-overlays) would eliminate this additional cost, but our analysis shows that the performance gain is reduced significantly.

[7] If the ISPs charge according to usage, then the cost of employing multiple ISP connections in the case of k-overlays may be higher or lower than the cost of using multiple connections in the case of k-multihoming.

While overlay routing significantly out performs using BGP paths via a single ISP, improving the routing flexibility of endpoint by allowing 2-3 ISP connections significantly reduces the gap between BGP-based routing and overlay routing.
The RTTs from multihoming to 3 ISPs are within 5–15% of those from employing a combination of overlay routing and multihoming. Throughput differences are more marginal with the best overlay path offers only 1–10% higher transfers speeds than the best multihoming path.
We also notice that multihoming offers similar, of not better, performance than 1-overlay routing. This implies that multihoming at the first hop is absolutely essential to overcome occasional serious problems in the access ISP.
The primary reason for the ability of overlay routing to offer better performance than multihoming (albeit only marginally so) is the ability of overlays to circumvent serious congestion along ISP paths that multihoming cannot avoid.
The superiority of overlay routing can be reduced even further if ISPs employed cooperative peering policies such as "cold potato" routing.
Multihoming route control cannot offer the near perfect availability of overlay routing. Nevertheless, multihoming can eliminate most failures observed by singly-homed end networks.

Table 5.4: Multihoming Route Control vs. Overlay Routing: Summary of key comparison results.

Deployment and operational overhead. Overlays and multihoming each have their unique set of deployment and performance challenges that our measurements do not highlight. Below, we consider the issues of ease of use and deployment, routing table expansion and routing policy violations.

Ease of use and employment. Overlay routing requires a third-party to deploy a potentially large overlay network infrastructure. Building overlays of sufficient size and distribution to achieve significantly improved round-trip and throughput performance is challenging in terms of infrastructure and bandwidth cost, as well as management complexity. On the other hand, since multihoming is a single endpoint based solution, it is relatively easier to deploy and use from an end-network's perspective.

Routing table expansion due to multihoming. An important overhead of multihoming that we did not consider in this study is the resulting increase in the number of routing table entries in backbone routers. ISPs will likely charge multihomed customers appropriately for any increased overhead in the network core, thus making multihoming less desirable. However, this problem occurs only when the stub network announces the same address range to each of its ISPs. Since ISPs often limit how small advertised address blocks can be, this approach makes sense for large and medium sized stub networks, but is more difficult for smaller ones. Smaller networks could instead use techniques based on network address translation (NAT) to avoid issues with routing announcements and still make intelligent use of multiple upstream ISPs [48]. This is discussed further in Chapter 6.

Violation of policies by overlay paths. One of the concerns that overlay routing raises is its circumvention of routing policies instituted by intermediate ASes. For example, a commercial endpoint could route data across the relatively well-provisioned, academic Internet2 backbone by using an overlay hop at a nearby university. While each individual overlay hop would not violate any policies (i.e., the nearby university node is clearly allowed to transmit data across Internet2), the end-to-end policy may be violated. While our analysis quantifies the number of routing policy violations, we did not consider their impact. Most Internet routing polices are related to commercial relationships between service providers. Therefore, it is reasonable to expect that the presence of an overlay node in an ISP network implies that the overlay provider and the ISP have some form of business agreement. This relationship should require that the overlay provider pay for additional expenses that the ISP incurs by providing transit to overlay traffic. Network providers would thus be compensated for most policy violations, limiting the negative impact of overlay routing.

Future changes to BGP. Thus far, we have discussed some important issues regarding overlays and route control in today's environment, but have not considered changes to BGP that may further improve standard Internet routing performance relative to overlays. For example, we only consider the impact of performance or availability-based route selection at the edge of the network. It is possible that transit ASes could perform similar route control in the future, thereby, exposing a superior set of AS paths to end networks. Another future direction is the development of new protocols for AS-level source-routing, such as NIRA [117], which allow stub networks greater control over their routes.

Limitations of the Comparison. Finally, we discussion the constraints imposed by our comparison study. Our observations may be constrained by a few factors such as the size of our testbed, the

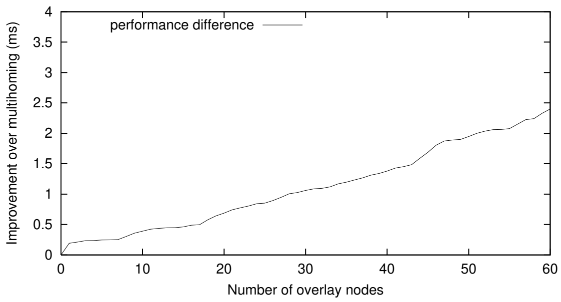


Figure 5.12: Impact of overlay network size on round-trip performance: This graph shows the mean difference between 3-overlays and 3-multihoming as overlay nodes are added.

coarse granularity of our performance samples, and our limited analysis of resilience. We discuss these issues in detail below.

Testbed size. In Figure 5.12 we compare the average RTT performance from 3-multihoming against 3-overlays, as a function of the number of intermediate overlay nodes available. The graph shows the RTT difference between the best 3-overlay path (direct or indirect) and best 3-multihoming path, averaged across all measurements as nodes are added one-by-one, randomly, to the overlay network. A different heuristic of adding nodes may yield different results. As the size of the overlay is increased, the performance of 3-overlays gets better relative to multihoming. Although the relative improvement is marginal, there is no discernible "knee" in the graph. Therefore it is possible that considering additional overlay nodes may alter the observations in our study in favor of overlay routing. However, the key point to notice from our measurements is that allowing endpoints the flexibility to select from overlay paths over 68 nodes did not, in the end, offer too much additional benefit over a choice between three paths. To widen the gap between overlay routing and route control by, say, about 30% or higher, we may require a much larger, geographically diverse overlay deployment. Such large overlays suffer from a few key drawbacks: they are expensive to deploy, manage and to subscribe to; and, they incur enormous overhead when tracking the performance of overlay paths (since the number of paths to probe scales quadratically in overlay size). Practical techniques for lowering measurement overhead in larger overlays are yet to be developed.

Granularity of performance samples. Our performance samples are collected at fairly coarse timescales (6 minutes intervals for round-trip time and 30 minutes for throughput). As a result, our results may not capture very fine-grained changes, if any, in the performance on the paths, and their effect on either overlay routing or multihoming route control. However, we believe that our results capture much of observable performance differences between the two path selection techniques for two key reasons: (1) our conclusions are based on data collected continuously over a week-long period, and across a fairly large set of paths, and (2) Zhang *et al.* observed that the "steadiness" of both round-trip time and throughput performance is at least on the order of minutes [118]. As we shall show in the next chapter, round-trip times on paths in our testbed have mean intervals of several minutes between changes of 30% or more. As such, we do not expect that a higher sampling frequency would yield significantly different.

Repair and failure detection. Our reliability analysis does not compare the relative ability of overlay routing and multihoming to avoid BGP convergence problems. For example, a peering link failure may affect routing between the peer ISPs until BGP re-converges. It is possible that some multihoming configurations cannot avoid such routing failures. We leave this comparison for future work.

The key observations in this comparison study can be simply summarized as follows: It is not necessary to circumvent BGP routing to achieve good end-to-end resilience and performance. These goals can be simply realized by enabling moderately higher route selection flexibility at end networks using purely endpoint based mechanisms.

So far, we considered the potential benefits of multihoming route control. We studied an ideal form of multihoming driven by several simplifying assumptions. In the next chapter, we ask whether these assumptions can be overcome in a practical deployment scenario while still preserving the performance benefits of multihoming route control.

Chapter 6

Practical Multihoming Route Control Strategies

Over the past few years, multihoming has been increasingly leveraged for improving wide-area network performance, lowering bandwidth costs, and optimizing the way in which upstream links are used [76], apart from ensuring resilience from service interruptions. Indeed, a number of products provide these route control capabilities both to large enterprise customers, which have their own public AS number and advertise their IP address prefixes to to upstream ISPs using BGP [106, 98, 53], as well as to smaller multihomed organizations which do not use BGP [79, 94, 35]. All of these products use a variety of mechanisms and policies for route control. However, very little is known about the exact mechanisms employed, or the quantitative benefits of the mechanisms.

In Chapter 4, we studied the potential performance benefits of multihoming route control mechanisms. We showed that performance could potentially improve by more than 25% in terms of RTT and 20% in terms of throughput when multiple upstream ISPs are employed. Furthermore, in Chapter 5, we showed that the performance benefits from multihoming are comparable with techniques such as overlay routing that allow much greater routing flexibility to end-networks. In either case, we evaluated an ideal form of multihoming where the end-network has perfect information about the performance across all ISPs at any time and could change routes arbitrarily often.

In this chapter, our goal is to understand if, and how, these benefits can be realized in a more practical multihoming scenario. Specifically, we address the following question:

> Can the benefits of multihoming route control be realized under practical deployment situations? If so, what exact mechanisms should end-networks employ?

We explore several design alternatives to realize the performance benefits from multihoming in practice. Our focus is on enterprise networks with multiple ISP connections. We focus primarily on mechanisms used for inbound route control, since enterprises are mainly interested in optimizing network performance for their own clients who download content from the Internet (i.e., sink data). However, as we discuss in this chapter, our mechanisms can also be extended to multihomed content provider networks which source more data than they sink.

We develop a variety of active and passive measurement strategies for multihomed enterprises to estimate the instantaneous performance of their ISP links and pick the best ISP for a given transfer.

We evaluate these strategies in the context of a NAT-based implementation to control the inbound ISP link used by enterprise connections. We address a number of practical issues such as the usefulness of past history to guide the choice of the best ISP link, the effects of sampling frequency on measurement accuracy, and the overhead of managing performance information for a potentially large set of target destinations. We evaluate these policies using several client workloads, and an emulated wide-area network where delay characteristics are based on a large set of real network delay measurements.

Our evaluation of the proposed schemes (for a 3-multihomed enterprise network) shows that both active and passive measurement-based techniques are equally effective in extracting the performance benefits of using multiple ISPs. These schemes offer about 15-25% improvement in Web response times (i.e., Web request completion times) when compared to using a single ISP. In deciding which ISP to select for a transfer, we show that the most current sample of the performance to a destination via a given ISP is a reasonably good estimator of the near-term performance to the destination. We also show that the overhead of collecting and managing performance information for various destinations is negligible. Finally, we conduct an initial study of mechanisms to control the ISP link used by external Internet clients who initiate connections to servers hosted in the enterprise.

Chapter outline. In Section 6.1, we describe our enterprise multihoming solution, various strategies for estimating ISP performance, and our route control mechanisms. Section 6.2 describes our implementation in further detail. In Section 6.3, we discuss the experimental set-up and results from our evaluation of the solution. Section 6.4 discusses additional route control design and operational issues. In Section 6.5, we summarize common techniques employed in commercial route control products as well as other related research studies. Finally, Section 6.6 summarizes the observations in this chapter.

6.1 Solution Overview

In order to realize the performance benefits of multihoming in practice, a route control solution requires three key functions (illustrated in Figure 6.1): (1) monitoring ISP links, (2) choosing the best ISP link at a given instant, and (3) directing traffic over the best ISP links. We discuss the functional design of each of these below. We discuss the actual implementation details in Section 6.2.

6.1.1 Monitoring ISP Links

Selecting the right ISP link over which to direct each transfer is crucial to realizing the performance benefits of multihoming from the enterprise network's perspective. The choice of the right ISP clearly depends on the time-varying performance of ISP links to the destination being accessed. There are two further issues in monitoring performance over ISP links: *what* to monitor and *how*.

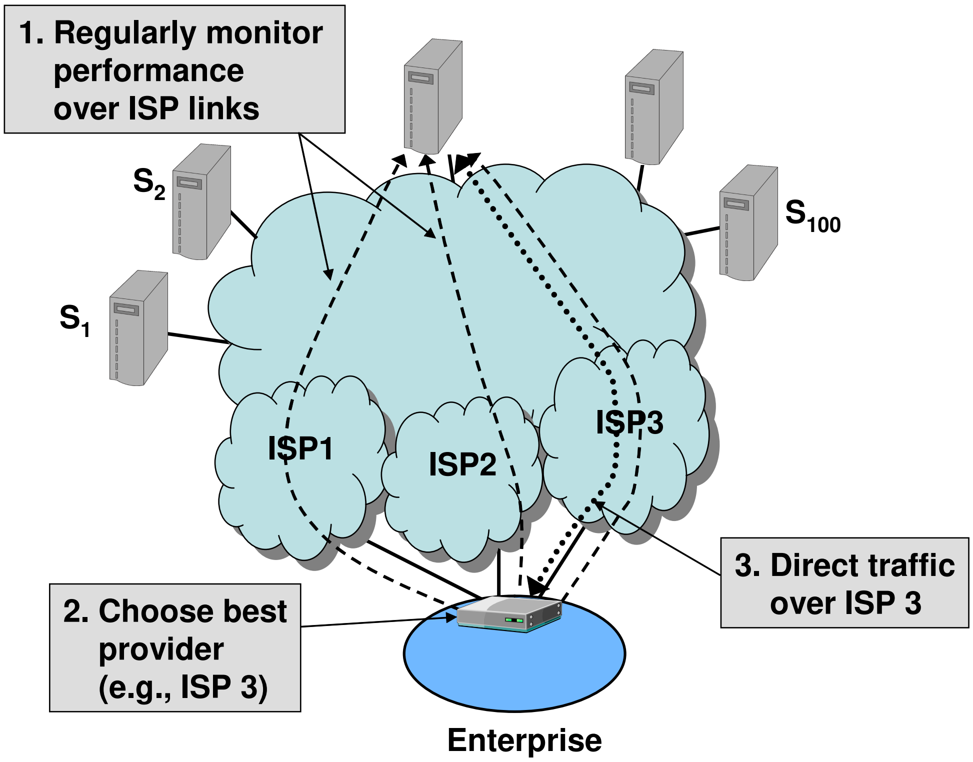


Figure 6.1: Solution steps: This figure illustrates the three main operations of an enterprise route control system.

An enterprise may ideally like to monitor the performance from all content provider sites over its ISP links. However, this may be infeasible, as large enterprises often access content from many different sources. A simple solution then is to monitor only the most important destinations, on the basis of the volume of requests made from the enterprise (e.g., the top 100 most frequently accessed destinations). This would ensure that a significant fraction of all flows will still experience good performance.

For the second question (i.e., how to monitor), two common approaches are active monitoring and passive monitoring. In active monitoring, the multihomed enterprise performs out-of-band measurements of performance to or from selected destinations across its ISP links. These measurements could be simple pings involving, for example, ICMP `ECHO_REQUEST` or TCP `SYN` packets to the destinations. Passive measurement mechanisms rely on observing the performance of ongoing transfers (i.e., in-band) to destinations, and using these observations as samples for estimating performance over an ISP link. However, in order to ensure that there are enough samples over all ISPs, it may be necessary to forcibly direct some transfers over particular ISPs.

An important issue in monitoring performance is the *time interval* for monitoring. A long interval between performance samples implies using stale information to estimate ISP performance. This might result in a suboptimal choice of the ISP link for a particular destination. While using smaller time intervals could address this issue, it could have a negative impact as well. In active monitoring, frequent measurements give rise to excessive measurement, wasted bandwidth and processing overhead. Some destinations might even interpret this traffic as a security threat. In passive monitoring, frequent sampling may cause too many connections to be directed over sub-optimal ISPs for obtaining "fresh" performance samples. Therefore, a careful choice of the interval size is important.

6.1.2 Choosing the Best ISP

The next component is to select the best ISP. This choice must be made on a per-destination basis, and at fine time-scales. An important question is whether (and how) historical data about ISP performance to a given destination should be used in determining the ISP to use. The historical performance of an ISP link to a destination can be tracked by keeping a smoothed, time-weighted estimate of the performance, for example an exponentially-weighted moving average (EWMA). If performance of using an ISP P to reach destination D at time t_i is s_{t_i} (as obtained from active or passive measurement) and the previous performance sample was from time t_{i-1}, then the EWMA metric at time t_i is:

$$\begin{aligned} EWMA_{t_i}(P, D) &= (1 - e^{-(t_i - t_{i-1})/\alpha}) s_{t_i} \\ &\quad + e^{-(t_i - t_{i-1})/\alpha} EWMA_{t_{i-1}}(P, D) \end{aligned}$$

where $\alpha > 0$ is a constant. A smaller value of α attaches less weight to historical samples. A value of $\alpha = 0$ implies no reliance on history. At any time, the ISP with the best performance as calculated above could be chosen for a given transfer. When no history is employed ($\alpha = 0$), only the most recent performance sample is used to evaluate the ISPs and to select the best.

6.1.3 Directing Traffic Over Selected ISPs

Once the best ISP for a transfer is identified, the traffic from the destination must be directed over the chosen link. Controlling the outbound direction of traffic is easy and well-studied. Our focus, rather, is on the *inbound route control* mechanism. Inbound control refers to selecting the right ISP or *incoming* interface on which to *receive* data. For an enterprise network, the primary mechanisms available are route advertisements and use of different addresses for different connections. Here, we discuss how these controls can be implemented.

If an enterprise has its own IP address block, it can advertise different address ranges to its upstream ISPs. Consider a site multihomed to two ISPs which owns a $/19$ address block. The site announces parts of its address block to its ISPs (e.g., a distinct $/20$ sub-block to each ISP). Then, depending on which of the two ISP links is considered superior for traffic from a particular destination, the site would use a source address from the appropriate $/20$ address block. This ensures that all incoming packets for the connection would arrive via the appropriate ISP link. In cases where the enterprise is simply assigned an address block by its upstream ISP, it may be necessary to also send outbound packets via the desired ISP to ensure that the ISP forwards the packets[1].

While ensuring that a connection uses a particular address lies at the heart of route control, different techniques must be employed for handling connections that are initiated from the enterprise, than for connections that are accepted into the site from external clients. These are discussed below.

Initiated Connections: Handling connections initiated from an enterprise site amounts to ensuring that the remote content provider transmits data such that the enterprise ultimately receives it over the chosen ISP. Inbound control can be achieved by having the edge router translate the source addresses on the connections initiated from its network to those belonging to the chosen ISP's address block (i.e., the appropriate $/20$ block in the example above) via simple NAT-like mechanisms. This ensures that the replies from the destination will arrive over the appropriate ISP.

Accepted Connections: Inbound route control over connections accepted into a site is necessary when the enterprise also hosts Internet servers which are accessed from outside. In this case, inbound control amounts to controlling the path (or the ISP link) on which a given client is forced to send request and acknowledgment packets to the Web server. This is not easy since predicting client arrivals and forcing them to use the appropriate server address is generally not possible.

However, techniques based on DNS or deploying multiple versions of Web pages can help to achieve inbound control for externally initiated connection. For example, the enterprise can use a different version of a base Web page for each ISP link. The hyperlinks for embedded objects in the page could be constructed with IP addresses corresponding to a particular ISP. Arriving clients would be given the appropriate base HTML page such that subsequent requests for the embedded objects arrive via the selected ISP. On the other hand, the essential function of the DNS-based technique is to provide the address of the "appropriate" interface for each arriving client. A preliminary study of the effectiveness of this approach is discussed in Section 6.4.

Our focus in the rest of the paper, however, is on the case of enterprise-initiated connections.

6.2 Implementation Details

We implement the above multihoming route control functions over a simple open source Web proxy called *TinyProxy* [112]. TinyProxy is a transparent, non-caching forward Web proxy that manages the performance of Web requests made by clients in moderate-sized, multihomed enterprises. Below, we present the details of our implementation of the three basic multihoming components in TinyProxy. For the sake of simplicity, we assume that the proxy is deployed by a multihomed end-network with three ISP links.

[1]In fact, like most enterprise route control products, we enforce outbound route control by transmitting packets to a destination along the same ISP as the one on which the traffic from the destination is received.

6.2.1 Performance Monitoring Algorithms

We implement both the active and passive measurement mechanisms, described in Section 6.1.1, for monitoring the performance of upstream ISP links.

6.2.1.1 Passive Measurement

The passive measurement module tracks the performance to destinations of interest by sampling ISP links using Web requests initiated by clients in the enterprise. The basic idea is to use new requests to sample an ISP's performance to a given destination, if the performance estimate for that ISP is older than the predefined sampling interval. If the module has current performance estimates for all links, then the connection is directed over the best link for the destination. The module maintains a performance hash table keyed by the destination (i.e., either the IP address or the domain name of the destination). A hash table entry holds the current estimates of the performance to the destination via the ISPs, along with an associated timestamp indicating the last time instant when performance to the destination via an ISP was measured. This is necessary for updating the EWMA estimate of historical performance.

Notice that without some explicit control, the hash table maintains performance samples to all destinations, including those rarely accessed. This may cause a high overhead of measurement, with connections to less popular destinations being all used up for obtaining performance samples. While maintaining explicit TTLs per entry might help flush out destinations that have not been accessed over a long period of time, it does not guarantee a manageable measurement overhead. Also, TTLs require maintaining a separate timer per entry, which is an additional overhead.

In view of this, we limit performance sampling to connections destined for the most popular sites, where popularity is measured in terms of aggregate client request counts. We achieve this in the following manner: Each hash entry also holds the number of *accesses* made to the corresponding destination. Upon receiving a connection request for a given destination, we update the access count for the destination using an exponentially weighted moving average (EWMA). The EWMA weight is chosen so that the access count for the destination is reset to $\sim$1 if it was not accessed for a long time, say 1 hour.

We use a hard threshold and monitor performance to destinations for which the total number of requests exceeds the threshold. We achieve this by looking for live entries in the table where the access counts exceed the threshold. In a naive hash table implementation for tracking the frequency counts of the various elements, identifying the popular destinations in this manner may take $O($hash table size$)$ time.

Other ways of tracking top destinations such as Iceberg Queries [38] or Sample-and-hold [34], may not incur such an overhead (see [20] for a survey of related approaches). Nevertheless, we stick with our approach for its simplicity of implementation. Also, as we will show later, the overhead from looking for the popular hash entries in our implementation is negligible. Note that this

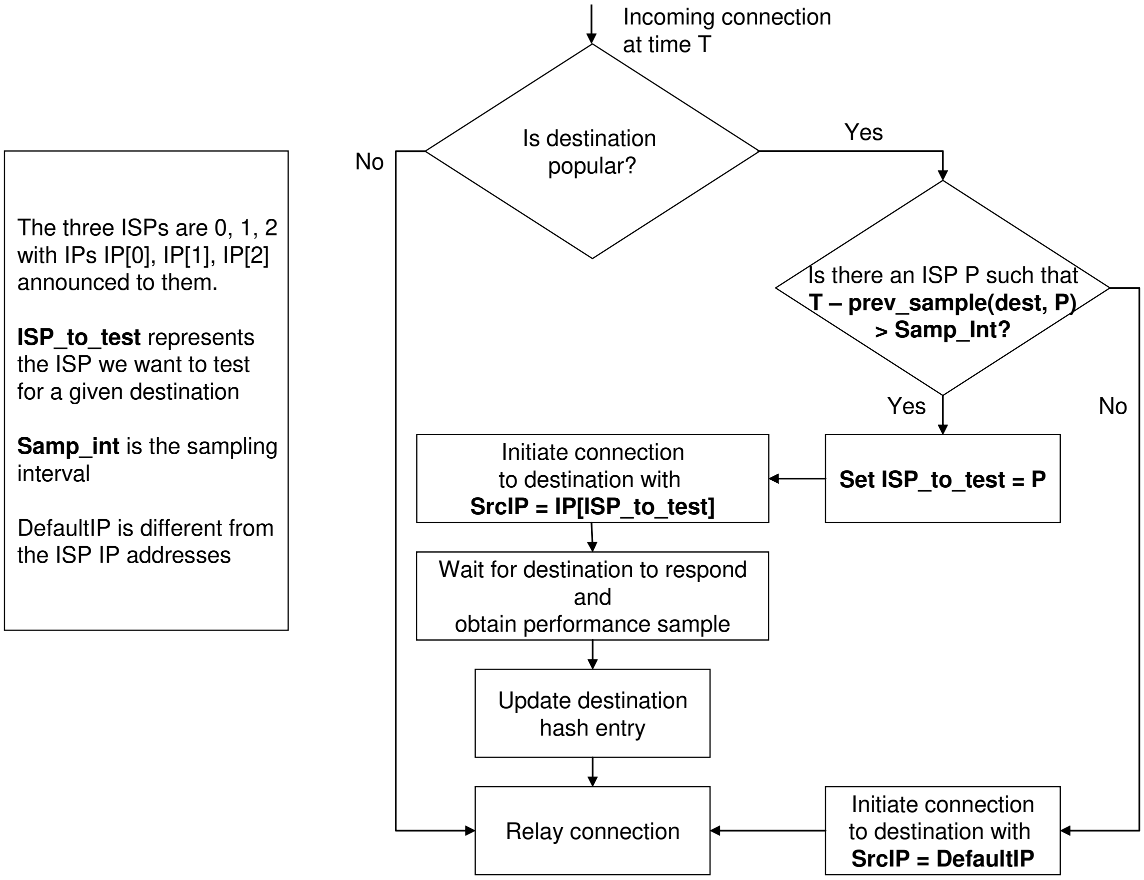


Figure 6.2: Monitoring ISP performance: The passive measurement scheme.

approach does not necessarily limit the actual number of popular destinations, for example in the relatively unlikely case that a very large number of destinations are accessed very often.

Figure 6.2 shows the basic operation of the passive monitoring scheme. When an enterprise client initiates a connection, the scheme first checks if the destination has a corresponding entry in the performance hash table (i.e., it is labeled popular). If not, the connection is simply relayed using an ISP link chosen in a random load-balancing fashion.

If there is an entry for the destination, the passive scheme scans the measurement timestamps for the three ISPs to see if the elapsed time since the last measurement on any of the links exceeds the predefined *sampling interval*. If so, the current connection is used to sample the destination along one such ISP links.

In order to obtain a measurement sample on an ISP link, the passive approach initiates a connection to the destination using a source IP address set such that the response will return via the link being sampled. Then, it measures the *turn-around time* for the connection, i.e., the time between the transmission of the last byte of the client HTTP request, and the receipt of the first byte of the

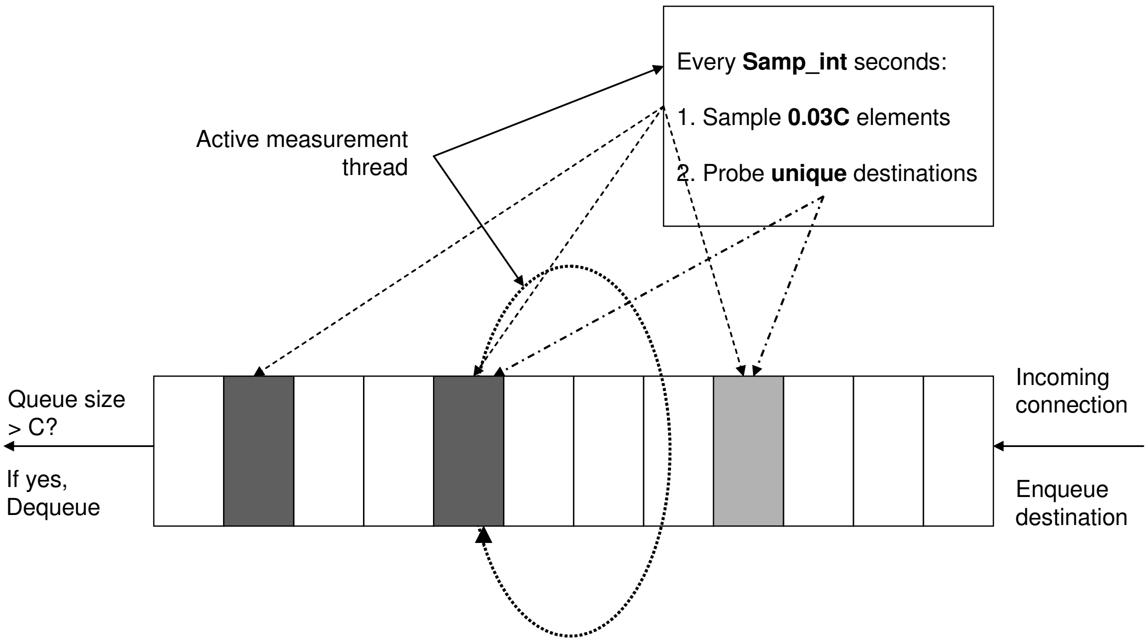


Figure 6.3: Monitoring ISP performance: The *SlidingWindow* active measurement scheme.

HTTP response from the destination. The passive scheme uses the observed turn-around time as the performance sample to the destination, and the corresponding entry in the hash table is updated using the EWMA method (Section 6.1.2). The remainder of the Web request proceeds normally, with the proxy relaying the data appropriately.

If all ISP links have current measurements (i.e., within the sampling interval), the proxy initiates a connection using the best link for the destination by setting the source IP address appropriately. This action is discussed further in Section 6.2.3.

6.2.1.2 Active Measurement

Similar to passive measurement, the active measurement scheme also maintains a hash table of the performance estimates to candidate destinations over the three ISPs. For active measurement, we use two techniques to identify which destinations should be monitored.

FrequencyCounts. Just like the passive measurement mechanism, in this scheme we track the number of client requests directed to each destination. Every T seconds (the sampling interval), we initiate active probes to those destinations for which the number of requests exceeds a fixed threshold.

SlidingWindow. This scheme maintains a window of size C that contains the C most recently accessed destinations (see Figure 6.3). The window is implemented as a fixed size FIFO queue, in which destinations from newly initiated connections are inserted. If this causes the number of elements to exceed C, then the oldest in the window is removed. Every T seconds (the sampling interval), an active measurement thread scans the window and chooses m% of the elements at random. After discarding duplicate destinations from this subset, the active-measurement scheme measures the performance to the remaining destinations along the ISPs.

The two active measurements schemes have their respective advantages and disadvantages. Both the schemes effectively sample the performance to destinations that are accessed more often relative to others. However, there are a few key differences. First, *FrequencyCounts* is deterministic since it works with a reasonably precise set of the popular destinations. *SlidingWindow*, on the other hand, may either miss a few popular destinations, or sample a few unpopular destinations. Second, *FrequencyCounts* in its simplest form, cannot easily track short-term shifts in the popularity of the destinations. These new, temporarily-popular destinations may not receive enough requests to exceed the threshold and force performance sampling for them. *SlidingWindow*, on the other hand, can effectively track small shifts in the underlying popularity distribution of the destinations.

Probe operation. Once a destination is selected for active probing, the active measurement scheme sends three probes, with different source IP addresses, corresponding to the three ISPs, and waits for the destination to respond. Since we found that a large fraction of popular Web sites filter ICMP `ECHO_REQUEST` packets, we employ a TCP-based probing mechanism. Specifically, we send a TCP `SYN` packet with the `ACK` bit set to port 80 and wait for an `RST` packet from the destination. We use the elapsed time as a sample of the turn-around time performance. We found that most sites respond promptly to the `SYN+ACK` packets.

When a response is received, we update the performance estimates to the destination for the corresponding ISP, along with the measurement timestamp. As described above, we update the performance estimate using an EWMA. If no response is received from a destination (which has an entry in the performance hash table), then a large positive value is used as the current measurement sample of the performance.

6.2.2 Switching ISPs

After updating all ISP entries for a destination in the performance hash, we switch to a new ISP only if it offers at least a 10% better RTT performance over the current best ISP. Since the hash entries are updated at most once every T seconds (both passive or active measurement schemes), the choice of best ISP per destination also changes at the same frequency.

6.2.3 NAT-based Inbound Route Control

Our inbound route control mechanism is based on manipulating NAT tables at the Web proxy to reflect the current choice of best ISP. We use the `iptables` packet filtering facility in the Linux 2.4 kernel to install and update NAT tables at the proxy. The NAT rules associate destination addresses with the best ISP link such that the source address on packets directed to a destination in the table are translated to an address that is announced to the chosen ISP.

For example, suppose ISP 1 is selected for transfers involving destination 1.2.3.4 and the addresses 10.1.1.1 was announced over the link to ISP 1. Then we insert a NAT rule for the destination 1.2.3.4 that (1) matches packets with a source IP of `defaultIP` and destination 1.2.3.4, and (2) translates the source IP address on such packets to 10.1.1.1.

Notice that if the NAT rule blindly translates the source IP on all packets destined for 1.2.3.4 to 10.1.1.1, then it will not be possible to measure the performance to 1.2.3.4 via ISP 2, assuming that a different IP address, e.g., 10.1.1.2, was announced over the link to ISP 2. This is because the NAT translates the source address used for probing 1.2.3.4 across ISP 2 (i.e., 10.1.1.2) to 10.1.1.1, since ISP 1 is considered to be the best for destination 1.2.3.4. To get around this problem in our implementation, we simply construct the NAT rule to only translate packets with a specific source IP address (in this case `defaultIP`). Measurement packets that belong to probes (active measurement) or client connections (passive measurement) are sent with the appropriate source address, corresponding to the ISP being measured.

6.3 Experimental Evaluation

In this section, we describe our experimental evaluation of the design alternatives proposed in Section 6.2. These include the performance of passive versus active monitoring schemes, sensitivity to various measurement sampling intervals, and the overhead of managing performance information for a large set of destinations. We focus on understanding the benefits each scheme offers, including the set of parameters that result in the maximum advantage.

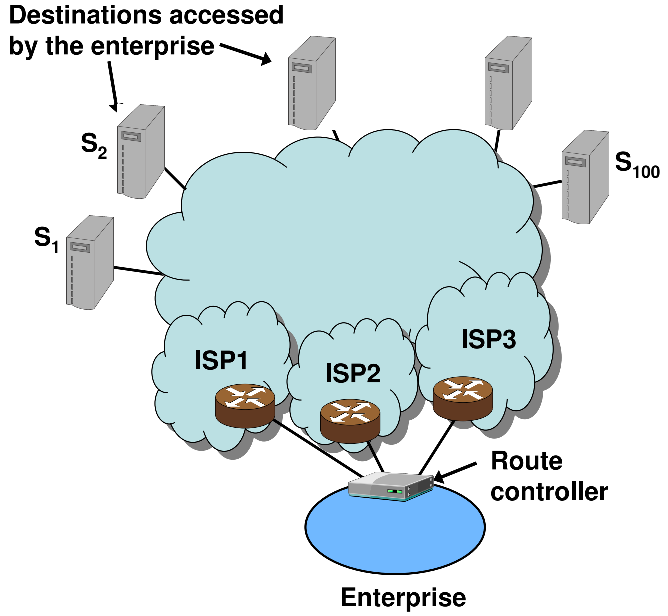
6.3.1 Experimental Set-up

We first describe our testbed setup and discuss how we emulate realistic wide-area network delays. Then we discuss key characteristics of the delay traces we employ in our emulation. Finally, we discuss the performance metrics we use to compare the proposed schemes.

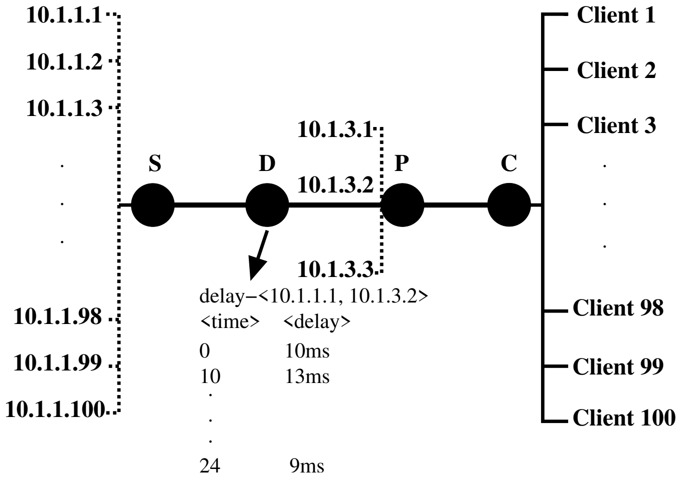
6.3.1.1 Testbed topology

We use the simple testbed topology shown in Figure 6.4(b). Our goal is to emulate a moderately-sized enterprise with three ISP connections and a client population of about 100 (shown in Figure 6.4(a)).

Node S in the topology runs a simple lightweight Web server and has one network interface configured with 100 different IP aliases—10.1.1.1 through 10.1.1.100. Each alias represents an instance of a Web server—10.1.1.1 being the most popular and 10.1.1.100 being the least popular.



(a) Multihomed enterprise



(b) Testbed topology

Figure 6.4: Testbed topology: The simple test-bed, shown in (b), is used to emulate the route control scenario shown in (a).

Node C runs 100 instances of clients in parallel. These clients make requests to the Web sites 10.1.1.1 through 10.1.1.100 as follows. The inter-arrival times between requests from a single client are Poisson-distributed with a mean of λ seconds. Notice that this mean inter-arrival rate translates into an average request rate of $\frac{100}{\lambda}$ requests per second at the server S. Each client request is for the i^{th} destination where i is sampled from the set $\{$10.1.1.1, $\ldots$, 10.1.1.100$\}$ according to a Zipf distribution with an exponent $\approx$ 2. In our evaluation, we set the parameters of the monitoring schemes (passive and active) so that the average rank of the destinations probed is 20, meaning that we explicitly track the top 40 most popular sites during each experiment. The object sizes requested by the client are drawn from a Pareto distribution with an exponent of 2 and a mean size of 5KB.

Node P in the topology runs TinyProxy. It is configured with one "internal" interface on which the proxy listens for connections from clients within the emulated enterprise. It has another interface with three IP aliases, 10.1.3.1, 10.1.3.2 and 10.1.3.3, representing addresses announced over the three ISP links. Node D is a *delay element*, running WaspNet [77], a loadable kernel module providing emulation of wide-area network characteristics on the Linux platform. We modify WaspNet to enforce packet delays (along with drops, and bandwidth limits) on a per-<source IP, destination IP> pair basis. We also modify it to support trace-based network delay emulation as illustrated in Figure 6.4(b).

In order to recreate realistic network delays between the clients and the servers in the testbed, we collect a set of wide area delay measurements using the Akamai content distribution network. We pick three Akamai server machines in Chicago, each attached to a unique ISP. We then run pings at regular intervals of 10s from each of them to 100 other Akamai servers located in various US cities and attached to a variety of ISPs. The measurements were taken over a one-day period on Dec 7th, 2003.

In this measurement, the three Akamai machines in Chicago collectively act as a stand-in for a multihomed network with three ISP connections. The 100 Akamai servers probed represent destinations contacted by end-nodes in the multihomed network. We use the series of delay samples between the three Akamai sources and the 100 destination servers as inputs to the WaspNet module to emulate delays across the ISP links.

6.3.1.2 Compressing time

It is quite time-consuming to emulate the entire day's worth of delays, multiple times over, to test and tune the parameters in the proposed scheme. One work-around could be to choose a smaller portion of the delay traces (e.g., 2 hours). However, a quick analysis of the delay traces we collected shows that there is little variation in the delays along the probed paths on a 2-hour timescale. Since our goal is to understand how effective our proposals are over a wide range of operating conditions, it is important to test how well the schemes handle frequent changes in the performance of the underlying network paths. With this in mind, we compress the 24-hour delay traces by a factor of 10, to 2-hour delay traces and use these as the real inputs to the WaspNet delay module. In these 2-hour traces, performance changes in the underlying paths occur roughly 10 times more often when compared to the full 24-hour trace. The characteristics of the 2-hour delay traces collected from the nodes in Chicago are shown in Table 6.1, column 2. We use these delay traces in our emulation.

We also want to ensure that the delays observed from the Chicago source nodes were not significantly different from typical delays experienced by a well-connected, multihomed network located

	Chicago trace	NYC trace	LA trace
Mean time between performance changes	79s	101s	105s
Standard deviation of time between changes	337s	487s	423s
Mean extent of performance change	$\pm 33\%$	$\pm 28\%$	$\pm 34\%$
Standard deviation of extent of change	$\pm 26\%$	$\pm 22\%$	$\pm 27\%$
Mean time between performance changes of 30%	298s	261s	245s

Table 6.1: Characteristics of the delay traces: Here "performance" refers to the delay along a given path.

in a major U.S. metropolitan area. Hence, we collect similar traces from source nodes located in two other cities, namely New York and Los Angeles. These traces were collected on March 20th, 2004. The statistics for these latter traces are shown in columns 2 and 3 of Table 6.1. These statistics show that the Chicago-based traces we use in our experiments have roughly the same characteristics as those collected at the other cities.

6.3.1.3 Comparison Metric

To evaluate the benefit from using our proposed route control schemes we compare the response time of transfers when using a particular scheme (i.e., $Resp_{(x,scheme)}$, for a transfer x), with the response time when the best of the three ISPs is employed for any transfer ($min_i\{Resp_{(x,ISP_i)}\}$):

$$\mathcal{R}_{scheme} = \frac{1}{||x||}\sum_x \frac{Resp_{(x,scheme)}}{min_i\{Resp_{(x,ISP_i)}\}} \tag{6.1}$$

Where, $||x||$ is the total number of transfers. In this chapter, we call $\mathcal{R}$ the "performance metric" or the "normalized response time" for the route control mechanism. The closer $\mathcal{R}$ is to 1, the better the performance of the scheme.

In the above computation, the response times from employing the best ISP any time (the terms in the denominator above) are computed in an offline manner, by forcing the transfers to use all the three ISPs, and selecting the ISP offering the best response time.

6.3.2 Experimental Results

We perform our experiments on the Emulab [33] testbed. We use 600MHz Pentium III machines with 256MB RAM, running Red Hat 7.3. We first describe how we select different client workloads

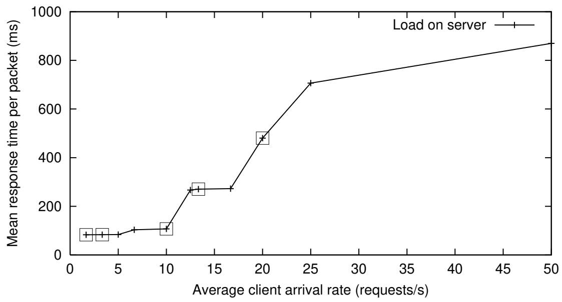


Figure 6.5: Web server load profile: Average response time in ms, per KB of the request, as a function of the average client arrival rate at the server in our topology (Figure 6.4(b)).

in our evaluation, and then move on to the evaluation of our route control strategies.

6.3.2.1 Selecting the Client Workloads

In Figure 6.5 we show the average response time per KB of client requests (i.e., the completion time for a request divided by the size of the request in KB), as a function of the average arrival rate of clients at the server S (i.e., $\frac{100}{\lambda}$ requests/s). The response time quickly degrades beyond an arrival rate of about 15 requests/s. After this point, it increases only marginally with the request rate. We select five different points on this load curve (highlighted), corresponding to arrival rates of 1.7, 3.3, 10, 13.3 and 20 requests/s , and evaluate the proposed schemes under these workloads. These workloads represent various stress levels on the server S, while also ensuring that it is not overloaded. The high variability in response times in overload regimes may impact the accuracy of our comparison of the proposed schemes.

In the remainder of this section, we focus on addressing the following questions:

1. To what extent do the route control schemes improve the performance of the multihomed site, relative to using the single best ISP alone?

2. Does employing historical samples help in better estimating future ISP performance?

3. How do active and passive measurement schemes compare in terms of the performance improvement they offer? Which of the two active measurement schemes—*SlidingWindow* or *FrequencyCounts*—works better?

4. At what time intervals should samples for ISP performance be collected?

5. What overheads do the proposed mechanisms incur?

The aggregate performance improvement from the passive measurement-based schemes is shown in Figure 6.6. Here, we set the EWMA parameter $\alpha = 0$ so that only the current measurement samples are used to estimate ISP performance, and select a sampling interval of 30s. The figure plots

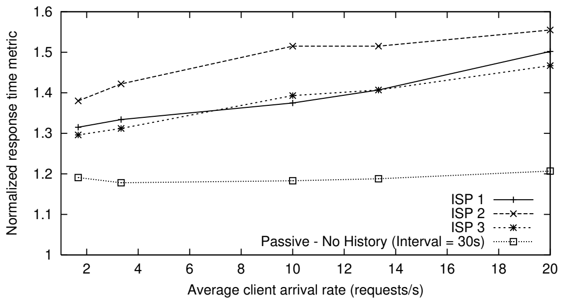
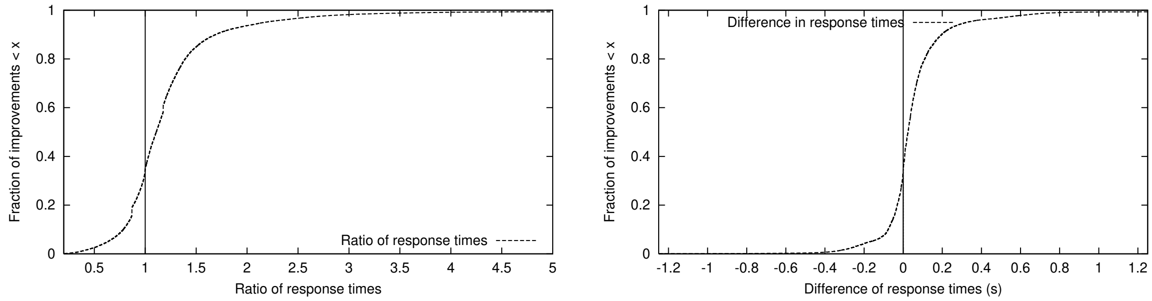


Figure 6.6: Performance improvement: The performance metric $\mathcal{R}$ for the passive measurement scheme with EWMA parameter $\alpha = 0$ (no history employed) and sampling interval of 30s. The graph also shows the performance from the three individual ISPs.



(a) Ratio of response times

(b) Difference in response times

Figure 6.7: Unrolling the averages: Ratio and the difference in the response times from using just ISP 3 for all transfers relative to using the passive measurement scheme. The average client arrival rate in either case is 13.3 requests/s.

the performance for the five client workloads. In addition, we show the performance from using the three ISPs individually.

The performance improvement relative to the best single ISP is significant – about 20-25% for the heavy workloads (right end of the graph) and about 10-15% for the light workloads (left end of the graph). The performance is still about 15-20% away from the optimal value of 1, however. The results for other sampling intervals (60s, 120s, 300s and 450s) are similar, and are omitted for brevity. The performance improvements from using the active measurement-based schemes are also similar and are discussed later.

6.3.2.2 Improvements from Route Control

Figures 6.7(a) and (b) illustrate the distribution of the response time improvements offered by the passive measurement scheme (for $\alpha = 0$ and sampling interval = 30s) relative to being singly-home to the best ISP from Figure 6.6, i.e., ISP 3. Figure (a) plots the CDF of the ratio of the response time

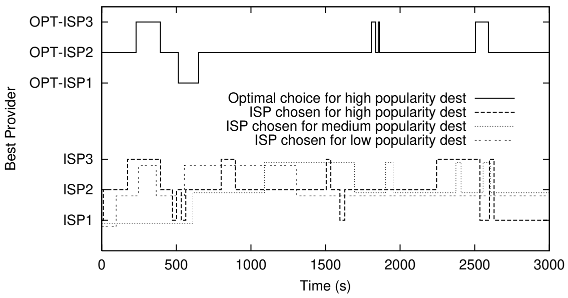


Figure 6.8: Route control at work: The ISPs chosen by the passive measurement-based route control scheme for destinations with different popularity levels.

from using ISP 3 to the response time from the passive measurement scheme across all transfers. These results are for the specific instance where the client arrival rate is 13.3 requests/s at the server. Figure (b) similarly plots the difference in the response times for the same client workload.

Notice, from either figure, that the passive measurement scheme improves the response time performance for over 65% of the transfers. Figure 6.7(a) shows that this route control scheme improves the response times by factors as large as 5 for a small fraction of transfers (about 1%), relative to being singly-homed. Similarly, Figure 6.7(b) shows that the scheme can improve response times by more than 1s for some transfers. Notice also that the passive measurement-based scheme ends up offering sub-optimal performance for about 35% of the transfers.

Figure 6.8 illustrates the operation of the passive measurement-based scheme. In this figure, we show the ISPs used over time for transfers to three different destinations—a popular destination (10.1.1.4), a moderately popular destination (10.1.1.16), and a less popular destination (10.1.1.38). Recall that the passive measurement-based scheme explicitly tracks the performance to the 40 most popular destinations. The sampling interval is 30s and the client arrival rate is about 13.3 requests/s.

From this figure, we note that changes to the route for the popular destination are made every 160s on an average. For the moderate and less popular destinations, the intervals are 300s and 550s respectively. For the passive scheme, the number of route changes depends on the popularity of the destinations—the more popular a destination is, the higher the frequency of its route changes. Figure 6.8 also shows the optimal choice of ISPs for the popular destination as a function of time, as determined from the underlying delay traces. Comparing this with the ISPs actually selected by the scheme, we sometimes see cases where our scheme makes a sub-optimal choice (e.g., between 750-800s, around 1500s, and 2250-2450s).

6.3.2.3 Employing History to Estimate Performance

Figure 6.9 plots the performance of the passive measurement scheme for three different values of the parameter α. These correspond to assigning 80%, 50% and 20% weight to the current measurement

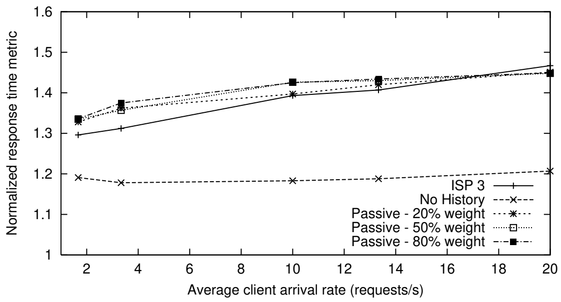


Figure 6.9: Impact of history: The performance achieved by relying on historical samples to varying degrees. These results are for the passive measurement-based strategy with a sampling interval of 30s.

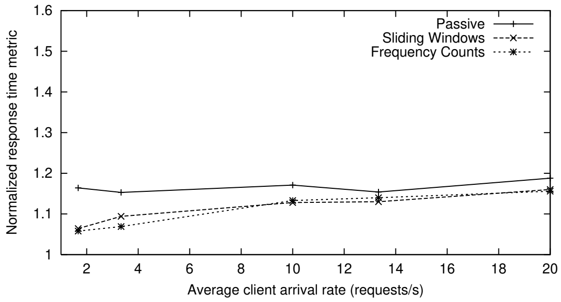
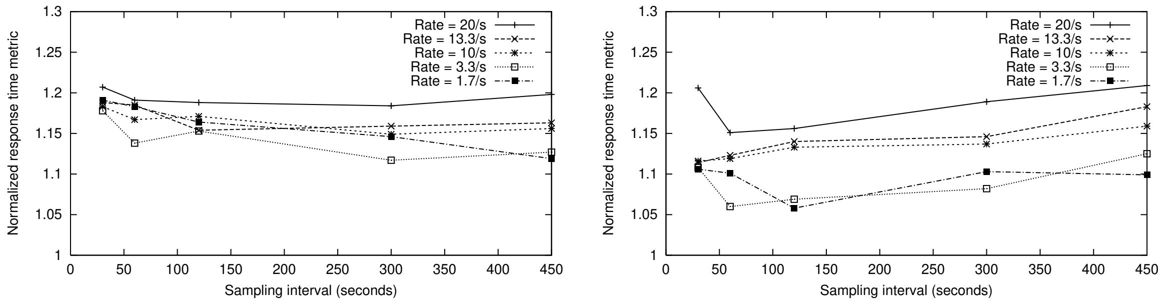


Figure 6.10: Active vs passive measurement: The performance of the two active measurement-based schemes, and the passive measurement scheme for a sampling interval of 120s.

sample and the remaining weight to the past samples. Although we only show results for a sampling interval of 30s, the performance from other interval sizes are similar. The figure also plots the performance when no history is employed ($\alpha = 0$) and the performance from using ISP 3 alone. Notice that the performance from employing history is uniformly inferior in all situations, relative to employing no history. In fact, historical samples only serve to bring performance close to that from using the single best ISP. These results show that the best way to estimate ISP performance is to just use the current performance sample as an estimate of near-term performance.

6.3.2.4 Active vs Passive Measurement

In Figure 6.10, we compare the performance from the two active measurement based techniques (i.e., *SlidingWindow* and *FrequencyCounts*) with the passive measurement approach. Since our earlier results showed that history does not help in improving performance, henceforth we present results in which no history is employed. We show the performance of the three schemes schemes for a sampling interval of 120s. Note that the two active measurement schemes offer comparable performance. However, the workloads we selected do not bring out other underlying trade-offs of these schemes. Figure 6.10 also shows that the active measurement-based schemes offer better



(a) Passive measurement

(b) *FrequencyCounts*

Figure 6.11: Impact of the sampling interval: The performance from using different sampling intervals from passive measurement-based and the *FrequencyCounts* active measurement-based schemes.

performance than the passive measurement scheme: moderate improvements of about 8-10% for the light workloads and slight improvement of 2-3% for the heavier workloads. This is expected, since the passive scheme uses existing transfers to obtain samples of performance across potentially sub-optimal ISP links.

6.3.2.5 Frequency of Performance Monitoring

Figure 6.11 shows the impact of the measurement frequency on the aggregate performance for the passive measurement scheme (Figure 6.11(a)) and the *FrequencyCounts* active measurement scheme (Figure 6.11(b)). From Figure 6.11(a) we notice that longer sampling intervals surprisingly offer slightly better performance for passive measurement. To understand this better, consider the curve for the client arrival rate of 10 requests/s. A client arrival rate of 10 requests/s implies that an average of $10T$ connections are made by the clients every T seconds, where T is the sampling interval. However, in order to obtain samples for a fraction f of the 100 destinations over the three ISPs, the passive measurement scheme will have to force $300f$ connections across the ISP links. This leaves a fraction $1 - \frac{30f}{T}$ which are not employed for measurement, and could be routed along the optimal ISP, assuming that the passive measurement yields reasonably accurate estimate of performance[2]. As T increases, the fraction of connections routed over the optimal path is likely to increase, resulting in a marginal improvement in performance. This explains the slight downward trend in Figure 6.11(a).

At the same time, infrequent sampling (i.e., large values of T) can have a negative impact on the overall performance. This can be seen in, Figure 6.11(b) which plots the performance from the *FrequencyCounts* scheme as a function of the sampling interval. A sampling interval of 450s suffers a 5-8% performance penalty relative to smaller intervals (e.g., 60s). In the case of *FrequencyCounts*, very aggressive sampling (e.g, an interval of 30s) could sometimes have a negative impact on the

[2] About a third of the connections employed for measurement can be expected to be routed along their optimal ISPs

	Passive	**Active FreqCount**	**Active SlidingWin**
Total performance penalty	18%	14%	17%
Penalty from inaccurate estimation only	16%	12%	14%
Penalty from measurement and NAT only	2%	2%	3%

Table 6.2: Analysis of performance overheads: Here "penalty" is defined as the value of $\mathcal{R} - 1$ in each case.

overall performance due to increased probing overheads at the proxy.

6.3.2.6 Analysis of overheads

As the performance results show, both passive and active measurement are still about 10% away from the optimal performance (specifically, the active measurement-based approaches). Three key factors contribute to this gap: (1) the accuracy of measurement techniques, and correspondingly, the accuracy of ISP choices, (2) overhead of conducting measurement, and (3) software overhead from making frequent updates to the NAT table[3] and enforcing NAT rules on most outgoing packets. In this section, we analyze the contribution of these factors on the performance of the proposed schemes.

To quantify the overhead of our implementation, we compare the performance due to the choices made by the route control proxy, with the performance when the best ISP choices are made in an offline manner for each connection. Recall that in order to compute the performance metric $\mathcal{R}$, we evaluated the response time of each ISP for every transfer offline so that the best ISP link for each connection was known, independent of the route control mechanisms (the terms in the denominator in Equation 6.1). If we combine these offline response time values with the decisions made by the proxy, we can estimate the performance penalty due to incorrect choices, independent of the software overheads (i.e., #2 and #3 above). The difference between the resulting performance metric, $\mathcal{R}$, and 1 gives us the performance penalty, not including overheads of the implementation.

We show the penalties from the three proposed schemes, obtained as stated above, in Table 6.2, row 2. The client arrival rate is 13.3 requests/s and the sampling rate is 30s. In this table, the numbers

[3] We could allow routes to change less frequently than the sampling interval, T, (e.g., every $T' > T$ seconds) but since we do not use performance history, this would be equivalent to sampling and updating every T' seconds.

in row 1 show the actual performance penalties suffered by the schemes in our implementation, taking all overheads into account (from Figure 6.11(a) and (b)). Notice that a large portion of the overall penalty is contributed by the inaccuracies in measurement and ISP selection (rows 1 and 2 are almost identical). Measurement and software overheads themselves result in a performance penalty of 2-3% (difference between rows 1 and 2, shown in row 3). As we mentioned earlier, active measurement-based techniques offer better overall performance. Also, the results above show that the measurement and route control overhead in active measurement schemes is small. This suggests that commercial route control products should prefer employing active measurement-based route control schemes to passive schemes.

6.4 Additional Design and Operational Issues

The route control mechanisms we presented here are a first attempt at understanding how to extract good performance from multiple ISP connections in practice. There are clearly a number of ways in which they can be improved. Also, we do not address several important issues, such as ISP costs and joint optimization of performance and reliability. Below, we briefly discuss some of these potential improvements and issues.

Handling lost probes. In our implementation of the active probing schemes, we send just one probe when collecting a performance sample for a (ISP link, destination) pair. It is possible that lost probes, e.g., due to transient congestion or even timeouts, may be misinterpreted for poor performance of the ISP path to the destination. This can cause unwanted changes in the ISP choice for the destination. We can prevent this by sending a short burst of, say, three probes per (ISP link, destination) pair. The performance reported by all three probes can then be used to estimate the quality of the ISP link, perhaps with a weighting to account for any observed losses.

Hybrid passive and active measurements. The accuracy of passive measurement can be improved by sending active probes immediately after a failed passive probe, for example when the observed connection ends unexpectedly. This increases confidence that the failed connection is due to a problem with the ISP link, as opposed to a transient effect.

In our implementation, paths to less popular destinations are not explicitly monitored (in both active and passive schemes). As a result, we may have to rely on passive observations of transfers to unpopular destination to ensure quick fail-over. For example, whenever the proxy observes several failures on connections to an unpopular destination, it can immediately switch the destination's default ISP to one of the remaining ISPs for future transfers.

Balancing performance and resilience. A key function of most route control products is to respond quickly to ISP failures. One of our findings in this study is that a relatively long sampling interval provides significant performance improvements. However, a long interval can slows down the end-network's reaction to path failures. Smaller sampling intervals can offer better resilience. For example, a sampling interval of 60s with active measurement works well in such cases, providing reasonably low overhead and good performance (Figure 6.11(b)), while ensuring a failover time of about one minute.

ISP pricing structures. In our study, we ignore issues relating to the the cost of the ISP links. Different ISP connections may have very different pricing policies. One may charge a flat price up to some committed rate, while another may use purely usage-based pricing or charge differently depending on whether the destination is "on-net" or "off-net." A more formal and thorough discussion of techniques for optimizing ISP usage costs as well as performance may be found in [45]. While we do not explicitly consider how to optimize overall bandwidth costs, we nevertheless believe that our evaluation of active and passive monitoring, and the utility of history, are central to more general schemes that optimize cost and performance together (such as the algorithms in [45]).

Long-lived TCP flows. In our route control schemes, an update to a NAT entry for a destination in the midst of an ongoing transfer involving that destination could cause the transfer to fail (due to the change in source IP address). We did not observe many failed connections in our experiments, and most of the flows were very short. However, this effect is nevertheless likely to have a pronounced impact on the performance of long-lived flows. It is possible to address this problem by delaying updates to NAT table until after ongoing large transfers complete. However, this increases the complexity of the implementation, as it involves identifying flow lengths, and checking for the existence of other long-lived flows at the time of update. It may also force other short flows to the same destination to traverse sub-optimal ISPs while the NAT update is delayed.

Impact on routing table sizes. Announcing small, non-aggregateable address sub-blocks to different upstream ISPs (Section 6.1.3) could affect the size of routing tables in the core of the network. This problem can be overcome if multihomed end-networks obtain *provider-assigned* addresses: instead of buying a single individual IP address block, an end-network simply acquires equal-sized independent address blocks from each of its ISPs. These address blocks could then be further aggregated by the ISPs.

Global effects of route control. Another important issue is the potential impact of the interactions between many enterprises deploying route control mechanisms. This will likely have an effect not only on the marginal benefits of the route control solutions themselves, but also on the network as a whole. However, a recent simulation-based study of this problem by Qiu et al. in [45] has shown that the the impact of multiple end-networks employing route control on any single multihoming user is very minimal, at the equilibrium of the interactions. Similarly, the impact, at equilibrium, on singly-homed users is also negligible. While these are positive observations, the issues of whether end-networks can reach an equilibrium, and, how stable the equilibrium is, still remain open.

About externally-initiated connections. Our implementation primarily considered handling connections initiated from within the enterprise, as these are common for current enterprise applications (e.g., to contact content providers). A route control product must also handle connections from outside clients to enable optimized access to servers hosted in the enterprise network. Recently, several route control device vendors have introduced features to use Domain Name System (DNS) resolution requests as a means to direct inbound client traffic over the desired link. Next, we describe preliminary measurements regarding the usefulness DNS-based approaches for externally-initiated connections. A more detailed description may be found in [85].

6.4.1 DNS for Inbound Route Control

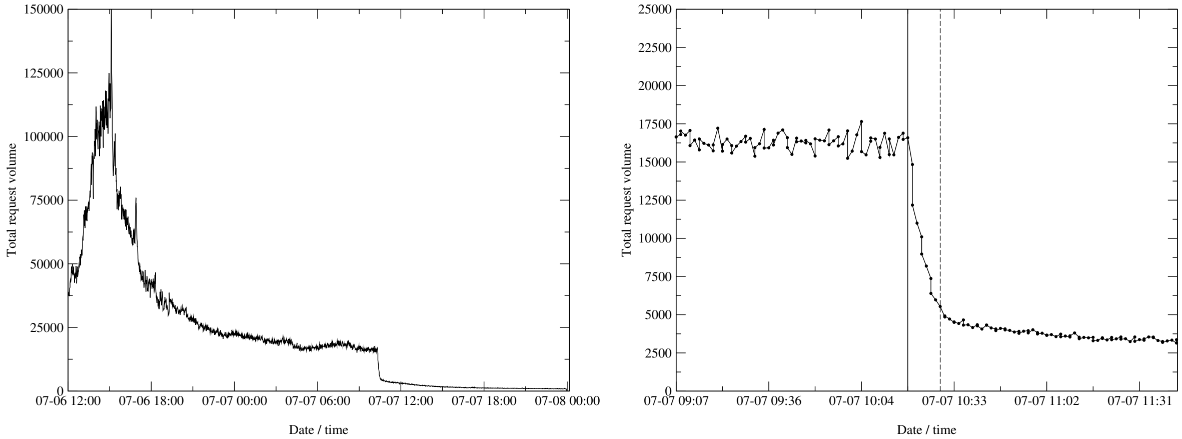
The destination IP address used by a remote client determines which ISP link is used for the connection request. Hence, by responding with the appropriate IP address when the client makes a request to resolve a service name (e.g., www.service.com), the inbound link can be selected. This is the key idea behind DNS-based route control and is very similar to using DNS as a server selection mechanism in content distribution networks [102].

While DNS is a convenient and relatively transparent mechanism, it is unclear whether it can respond quickly enough for dynamic route control. Responses to name resolution requests have an associated time-to-live (TTL) value that determines how long the response should be cached by the client's local name server. Ideally, by setting the TTL to a very small value (e.g., 10s or even zero), it is possible to force external clients to resolve the IP address frequently, thus providing fast responsiveness. In practice, however, this is complicated by the behavior of the wide variety of applications and DNS servers deployed in the Internet. Many applications perform their own internal DNS caching that does not adhere to the expected behavior, and some older implementations of DNS software have been reported to ignore low TTL values. These artifacts make it difficult to predict how quickly clients will respond to changes communicated via DNS responses.

In order to quantify the responsiveness of DNS in practice, we perform a simple analysis of client behavior in response to DNS changes during a large Web event. We collect logs from a set of Web caches that served requests for content related to a Summer 2003 sporting event with global audience. During the event, when the request rate was very high, the authoritative name servers directed all clients to the set of caches with a 10min TTL. After the event, the name servers are updated to direct clients to lower capacity origin servers. Ideally, all traffic to the caches should subside after 10min.

Figure 6.12 shows the aggregate request volume to all caches over time, just before and after the DNS change, where requests were gathered into 1-minute intervals. During the one-hour period after the DNS change, requests came from about 13400 unique client IP addresses and 5600 unique IP subnets. The number of subnets is computed by clustering client IP addresses using BGP tables obtained from [60, 114].

Figure 6.12(a) shows the last part of the trace, with a clear peak occurring on the last day of the event, followed by a period of relatively constant and sustained traffic, and finally a sharp drop-off corresponding to the time when the DNS is updated. Figure 6.12(b) focuses on the time around the



(a) 2-day trace (b) 2-hour portion

Figure 6.12: DNS responsiveness: This figure shows traffic volume over time just before and after a DNS change. The left graph (a) shows a 2-day period around the end of the event, while (b) focuses on a 2-hour period around the time of the DNS update.

DNS update; the solid line denotes the time of the update and the dashed line is the time when the 10min TTL expires. Between these times, the request volume decreases by 66%. The remaining third of the traffic decays very slowly over a period of more than 12 hours. While this analysis is not definitive, it does suggest that DNS is, at best, a coarse-grained mechanism for controlling traffic.

6.5 On Common Route Control Practices

Our focus in this chapter was on improving Internet performance of enterprise networks via route control. Several research studies and products have considered other benefits of multihoming route control. For example, in [48, 94], Guo et. al conduct trace-driven experiments to evaluate several design options using a commercial multihoming device. They evaluate the ability of several algorithms to balance load over multiple broadband-class links to provide service similar to a single higher-bandwidth link. The authors find that the effectiveness of hash-based link selection (i.e., hashing on packet header fields) in balancing load is comparable to load-based selection. In addition, their results show that managing load at a connection-level granularity is only slightly less effective than per-packet load balancing. Andersen et al. similarly consider various mechanisms for improving the reliability of Web access for DSL clients in [15].

A number of vendors have recently developed dedicated networking appliances [92, 35, 79] or software stacks [107, 93] for optimizing the use of multihomed connectivity in enterprises settings where BGP is not used. Most of these products use techniques similar to the ones we evaluate in our study. However, their focus is geared more toward balancing load and managing bandwidth costs across multiple ISP links, rather than optimizing performance. All of these use NAT-based control of inbound traffic and DNS to influence links used by external client-initiated connections. They also ensure, by tracking sessions or using policy-based routing, that the same ISP link is used in both directions.

Another class of products and services are targeted at settings where BGP is employed, such as large data centers or campuses [98, 103]. These products mainly focus on outbound control of routes and, as such, are more suited for content providers which primarily source data. Details of the algorithms used by any of the above commercial products to monitor link performance or availability are generally proprietary, and little information is available on specific mechanisms or parameter settings. Here, we review the general approaches taken in enterprise route control products.

Most products employ both ICMP ping and TCP active probes to continuously monitor the health of ISP links, enabling rapid response to failure. In some cases, hybrid passive and active monitoring is used to track link performance. For example, when a connection to a previously unseen destination is initiated from an enterprise client, active probes across the candidate links sample performance to the destination. Connections to known destinations, on the other hand, are monitored passively to update performance samples. Another approach is to use active probing for monitoring link availability, and passive monitoring for performance sampling. Some products also allow static rules to dictate which link to use to reach known destinations networks.

Finally, some products and recent research studies [15] have suggested using "race"-based ISP performance measurements, wherein SYN packets sent by enterprise clients to initiate connections are replicated by the route control device on all upstream ISPs (using source NAT). The link on which the corresponding SYN-ACK arrives from the server first is used for the remainder of the connection. The route control device sends RST packets along the slower paths so that the server can terminate the in-progress connection establishment state. The choice of best link is cached for some time so that subsequent connections that arrive within a short time period need not trigger a new race unless a link failure is detected. It is easy to extend the active and passive probing techniques presented in this chapter to incorporate "race"-based performance optimization. We believe that this approach will further enhance the performance improvements from the route control techniques discussed in this dissertation.

6.6 Summary of Observations and their Implications

Our goal in this chapter was to experimentally evaluate a variety of practical mechanisms and policies for realizing performance benefits from ISP multihoming. We focused on the scenario of multihomed enterprises that wish to leverage multiple ISPs to improve the response time performance for clients downloading content from Internet Web servers. Using a real Linux-based route control implementation and an emulated wide-area network testbed, we evaluated several design alternatives. These included the performance of passive versus active monitoring schemes, sensitivity to various measurement sampling intervals, and techniques to manage performance information for a large set of destinations.

The key findings from our evaluation of a 3-multihoming scenario are summarized in Table 6.3. Our evaluation shows that both active and passive measurement-based route control schemes offer significant performance benefits in practice, between 15% and 25%, when compared to using the single best-performing ISP. The performance from these schemes is within 5–15% of the optimal possible benefits. For light to moderate client workloads the performance from active measurement schemes was much better. Our experiments also show that using historical performance to choose the best ISP link is not necessary: The most current measurement sample gives a good estimate. We showed that the performance penalty from collecting and managing performance data across various destinations is negligible.

To summarize, the key contribution of our route control implementation is to shows that the benefits of multihoming route control can be realized in practice using very simple mechanisms, e.g., active and passive probing, and NAT. In effect, we showed that increasing the Internet routing flexibility of end-networks by modest amounts using purely end network-based mechanisms substantially improves their Internet performance.

So far, we have shown how multihoming route control can effectively improve the Internet performance of endpoints in the network today. However, as more and more endpoints employ higher-speed access links, and as the Internet grows in its size, the volume of traffic as well as the application mix in the network will change drastically from what we observe today. An important question to ask, then, is whether techniques such as multihoming route control can be effective in the future Internet. We address this issue in the next chapter.

The route control schemes we describe can significantly improve client Web response times at a multihomed site by up to 25% in our experiments.
The performance from our proposed route control mechanisms is about 10% away from the optimal benefits, on average.
Relying on historical samples to monitor performance of ISPs (e.g., using EWMA) is not very useful, and sometimes may be detrimental to performance. The most current measurement sample is a very good estimator of near-term performance of an ISP link.
Both passive and active measurement-based schemes better performance than using single ISP connections. However, the latter approach offers substantially better performance for light to moderate client workloads.
The overhead introduced by aggressive performance sampling may slightly reduce the overall performance benefit of route control schemes. A sampling interval of one minute, on the other hand, seems to offer good performance-overhead trade-offs.
The overhead from measurements (in active measurement schemes) and frequent updates to the NAT table are negligible. Most of the performance penalties arise from the inaccuracies of the measurement and estimation techniques.
DNS is not an effective mechanism to achieve inbound control of requests for externally initiated connections.

Table 6.3: Practical Multihoming Route Control: Summary of key observations regarding route control implementation.

Chapter 7

Scaling of Congestion in the Internet

The Internet grows in size every day. As time progresses, more end-hosts are added to the edge of the network. Correspondingly, to accommodate these new end-hosts, ISPs add more routers and links. History has shown that the addition of these links maintains certain macroscopic properties of the Internet. For example, the interconnection between ISPs in the Internet continues to follow a power law graph structure [37]. Also, the addition of new end-hosts over time places a greater load on the network as a whole. Fortunately, the improvement of network technology operates over the same time period. We expect the network links at the edge and core of the network to improve by a similar performance factor, since they both typically follow similar Moore's Law-like technology trends.

Unfortunately, due to the topology of the network and behavior of Internet routing, the increase in load may be different on different links of the network. As a result, it may be necessary for the speed of some key "centrally located" hot-spot links in the network to improve much more quickly than others. If this is true, then these key parts of the network will eventually emerge into persistent bottlenecks. Under these circumstances, routing-based mechanisms can no longer be employed by end-networks to improve their performance. Consider route control, for example. Multihoming paths will still have to traverse links in tier-1 ISPs (due to the global reach of the ISPs). Similarly, for most placements of overlay nodes, even overlay paths will traverse tier-1 networks (although the likelihood of this happening is lower compared to route control). Therefore, if links belonging to such backbone carriers emerge as hot-spots, neither approach can help end-networks overcome these inevitable hot-spots. We can then say that the network has poor scaling properties. In such a situation, we would either need to drastically adjust the Internet's routing behavior or change the structure of the network (for example, the interconnections between ISPs) to ensure good future performance.

On the other hand, if the worst congestion scales well with the network size then we can expect the network to continue to operate as it does now. Also, routing-based mechanisms will continue to be effective at improving endpoint performance. In this chapter, we perform a preliminary study of the scaling properties of the Internet. Using reasonably realistic theoretical models of network evolution and inter-domain routing, we seek to answer the following question:

How does the maximum congestion in the Internet scale with the network size?

Our analysis focuses on the Internet *AS-level* interconnection. We consider a model of network evolution based on Preferential Connectivity [21], and a simple model of traffic in which a unit amount of flow between every pair of nodes is routed along the shortest path between them. We employ simple combinatorial/probabilistic arguments to give bounds on the maximum congestion in the AS-level graph. We also conduct simulations of the congestion on the links in the network, based both on real and on synthetically generated *AS-level* topologies and synthetic traffic matrices. Through our simulations, we also investigate the impact of several key factors on the worst congestion in the network, such as variants of the inter-domain routing algorithm, alternate traffic matrices, and finally, alternate topologies for the AS-level interconnection.

The key result in this chapter is that the maximum congestion in Internet-like graphs scales poorly with the growing size of the graph. Specifically, the maximum congestion for shortest path routing and uniform traffic matrices is worse than $n^{1+\Omega(1)}$, with the exponent depending on the degree of "skew" in the power law degree distribution of the graph[1]. Our simulations show that policy routing in the AS graph results in roughly the same maximum congestion as shortest path routing, but certainly not worse. When alternate, non-uniform traffic models are considered, the congestion scaling properties of power law graphs worsen substantially. We also show that in terms of the maximum congestion, power law trees are considerably worse than power law graphs. In contrast, graphs with exponential degree distribution have very good congestion properties.

Further, we also discuss simple guidelines to change the ISP-level interconnections that result in a dramatic improvement in the congestion scaling properties of Internet-like graphs. We show that when parallel links are added between adjacent nodes (i.e., ASes) in the network according to simple functions of their degrees or the number of neighboring ASes (e.g., the minimum of the two degrees), the maximum congestion in the resulting graph scales linearly. This heuristic for adding edges reflects the desired amount of peering between neighboring ASes in the Internet in order to guarantee good overall congestion scaling properties.

Chapter outline. In Section 7.1, we formalize our analytical approach and discuss our simulation set-up. The analysis is presented in Section 7.2. Section 7.3 presents the results from our simulations. In Section 7.4, we discuss the implications of our results on network design and present mechanisms to alleviate the poor congestion scaling of the Internet graph. In Section 7.5, we survey past work on modeling the Internet and results for deriving congestion scaling properties of general

[1]There is some disagreement about whether a power law correctly models the degree distribution of the Internet graph. However, it is widely agreed that the distribution is heavy-tailed. While our main results (specifically, simulation results) focus on power law distributions, we believe that they hold equally well for other such heavy-tailed distributions (e.g. Weibull).

graphs. Finally, in Section 7.6, we summarize the key observations in this chapter and present a few caveats of our approach.

7.1 Methodology

We first give a precise formulation of the problem, laying out the key questions we seek to address via analysis. We also describe the simulation set-up for corroborating and extending our analytical arguments.

7.1.1 Problem Statement

Let $G = (V, E)$ be an unweighted graph, representing the Internet AS-level graph, with $|V| = n$. Let d_v denote the total degree of a vertex v in G. We are given three key aspects pertaining to the graph G: the degree distribution of the graph, the routing algorithm used by the nodes in the graph to communicate with each other and the traffic demand matrix determining the amount of traffic between pairs of nodes in the graphs. We give precise definitions of these three aspects, in turn, below.

We will mostly be concerned with graphs having a power law degree distribution, defined below[2].

Definition 1 *We say that an unweighted graph G has a power law degree distribution with exponent α, if for all integers d, the number of nodes v with $d_v \geq d$ is proportional to $d^{-\alpha}$.*

Similarly, graphs with exponential degree distribution are those in which the number of nodes v with $d_v \geq d$ is proportional to $e^{-\beta d}$, for all integers d. Henceforth, we will refer to such graphs as power law graphs and exponential graphs respectively.

Let $\mathcal{S}$ denote a routing scheme on the graph with $\mathcal{S}_{u,v}$ representing the path for routing traffic between nodes u and v. We consider two different routing schemes:

1. **Shortest Path Routing:** In this scheme, the route between nodes u and v is given by the shortest path between the two nodes in the graph G. This reflects route selection in BGP where every AS prefers paths with the least number of ASes. When there are multiple shortest paths, we consider the maximum degree of nodes along the paths and pick the one with the highest maximum degree. This tie-breaking rule is reflective of the typical policies employed in the Internet—higher degree nodes are typically much larger and much more well-provisioned providers than lower degree nodes and are in general used as the primary connection by

[2]There is some disagreement about whether a power law correctly models the degree distribution of the AS-level graph. However, it is widely agreed that the distribution is heavy-tailed. While our main results (specifically, simulation results) focus on power law distributions, we believe that they hold equally well for other such heavy-tailed distributions (e.g. Weibull).

stub networks. In Section 7.3.3, we consider alternate tie-breaking schemes such as random choice and favoring lower degree nodes, and show that the tie-breaking rule does not affect our results.

2. **Policy Routing:** In this scheme, traffic between nodes u and v is routed according to BGP-policy. We classify edges as peering edges or customer-provider edges (that is, one of the ASes is a provider of the other). These commercial relations between ISPs are known to give rise to "valley-free" routing, in which each path contains a sequence of customer to provider edges, followed by at most one peering edge, followed by provider to customer edges. The key difference between policy routing and shortest path routing, then, is that in the former case, we are restricted to choosing between multiple shortest path routes which are all compliant with valley-free routing. Our goal in studying these two forms of routing is to understand if commercial policies between ISPs significantly change our observations regarding network congestion.

A traffic vector τ is a vector containing $\binom{n}{2}$ non-negative terms, with the term corresponding to (u, v) signifying the amount of traffic between the nodes u and v. The congestion on an edge e due to traffic vector τ and routing scheme $\mathcal{S}$ is given by the sum of the total amount of traffic that uses the edge e: $\mathcal{C}_{\tau,\mathcal{S}}(e) = \sum_{(u,v):e \in \mathcal{S}_{u,v}} \tau(u, v)$.

We define the edge congestion due to traffic vector τ and routing scheme $\mathcal{S}$ to be the maximum congestion on any edge in the graph:

$$\text{EDGE-CONGESTION}_{\tau,\mathcal{S}}(G) = \max_{e \in E} \mathcal{C}_{\tau,\mathcal{S}}(e)$$

Our goal is to quantify the congestion in a graph with power law degree distribution, for shortest path and policy routing schemes, due to different traffic vectors. Specifically, we consider the following three traffic vectors:

1. **Any-2-any:** This corresponds to the all 1s traffic vector, with a unit traffic between every pair of ASes. While very simplistic, this model is amenable to analysis and provides a baseline for comparison against more complex traffic vectors.

2. **Leaf-2-leaf:** In order to define this model, we classify nodes in the graph as *stubs* and *carriers*. Stubs are nodes that do not have any customers. In other words, consider directing all customer-provider edges in the graph from the customer to the provider. Peering edges are considered to be bi-directed edges. Then, vertices with no incoming edges (corresponding to ASes with no customers) are called stubs or leaves in the graph. In this model, there is a unit of traffic between every pair of stubs in the graph. No other AS sources or sinks traffic on its own.

3. **Clout:** This model approximates the fact that "well-placed" sources, that is, sources which have a high degree and are connected to high degree neighbors, are likely to transmit larger

amounts of traffic than other sources. Accordingly, in this case, $\tau(u, v) = f(d_u, c_u)$, where u and v are stubs, c_u is the average degree of the neighbors of u and f is an increasing function. As in the previous case, there is no traffic between nodes that are not stubs. In what follows, we only use the function $\tau(u, v) = f(d_u, c_u) = d_u c_u$ for stubs u, v. This is the most sophisticated of the traffic vectors we consider.

Admittedly, our choice of the models for Internet routing, as well those for Internet traffic matrices, are somewhat unrealistic. We do try to model real phenomena, such as popularity of some ASes, but make no claims to the realism of our models. However, we still use them in our analysis for reasons of simplicity and for lack of realistic Internet-wide traffic traces.

7.1.2 Simulation Set-up

Our simulations serve two purposes: (1) to corroborate our theoretical results, and, (2) to characterize the congestion in more realistic network models than those considered in our analysis.

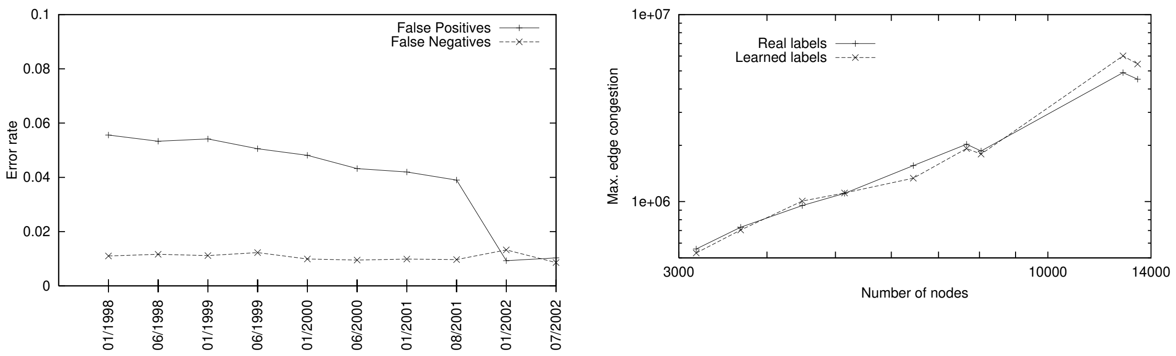
Our simulations are run on two different sets of graphs. The first set of graphs contains maps of the Internet AS topology collected at 6 month intervals between Nov. 1997 and April 2002, available at [78]. The number of nodes in any graph in this set is at most 13000, the maximum corresponding to the April 2002 set. The second set of graphs contains synthetic power law graphs generated by Inet-3.0 [115]. In this set, we generate graphs of sizes varying from $n = 4000$ to 50000 nodes [3].

As mentioned earlier, in order to implement the leaf-2-leaf and clout models of communication, we need to identify stubs in the network (note that these might have a degree greater than 1). Additionally, in order to implement policy routing, we need to classify edges as peering or non-peering edges. To do this for the real AS graphs, we employ the relationship inference algorithms of Gao [42] to label the edges of the graphs as peering or customer-provider edges. These algorithms use global BGP tables [23] to infer relationships between nodes. Then, we use these relationships to identify stubs, as nodes that are not providers of any other node. Henceforth, we shall refer to the real AS graphs as *accurately labeled real* graphs (ALRs).

Labeling edges and identifying stubs in the synthetic graphs of Inet is more tricky since we do not have the corresponding BGP information. We will refer to synthetic graphs, labeled using the algorithms described below, as *heuristically labeled synthetic* graphs (HLSs). We use different algorithms for classifying nodes (this is key to implementing leaf-to-leaf communication) and edges (this is key to implementing policy routing in synthetic graphs). We discuss these next.

Stub identification. We identify stubs in synthetic graphs as follows: For any edge $e = (v_1, v_2)$, we assign v_1 to be the provider of v_2 whenever $degree(v_1) \geq degree(v_2)$. Notice that we do not explicitly identify peering edges (although edges between nodes of identical degree will be bidirectional). We then identify stubs in graphs labeled as above.

[3] In all our simulations, for any metric of interest, for each n, we generate 5 different graphs of n nodes (by varying the random seed used by the Inet graph generator) and report the average of the metric on the 5 graphs.



(a) Accuracy of Stub Identification

(b) Accuracy of Edge Labeling

Figure 7.1: Accuracy of heuristics: The graph on the left shows the accuracy of our simple stub identification algorithm. The graph on the right shows the error in the maximum congestion due to our machine-learning based edge-classification algorithm.

We test the accuracy of this stub-identification algorithm on real AS graphs by comparing the labels produced by our algorithm to the true labels of ALRs, and compute the fraction of false positives and false negatives[4] in these. The accuracy results are shown in Figure 7.1. Note that our simple algorithm has a very low error rate. Notice that the inference algorithms of Gao [42] have some error intrinsically and hence some of the labels on the ALRs might actually be inaccurate.

Edge classification. Simply considering all edges in the graph to be customer-provider edges, as done above, is not useful for the purposes of edge classification. Specifically, it results in a significant error on the maximum congestion in real graphs employing policy routing (results omitted for brevity). In order to improve the accuracy of labeling edges, we resort to machine learning algorithms. However, coming up with a good machine learning algorithm for the classification is a challenging task, because there is a huge bias toward customer-provider edges in the graphs (roughly 95% of the edges are customer-provider edges). We use the 3-Nearest Neighbor [75] algorithm for classifying edges as peering or non-peering: each edge in the unlabeled graph is classified as a peering edge if among the three edges most similar to it in the labeled graph, at least two are peering edges. Similarity between edges is judged based on the degrees of their respective end points and neighboring vertices. We measure the accuracy of the procedure by applying it to real graphs and then comparing the classification with true labels.

Our machine learning algorithm gives only 20% accuracy on peering edges and about 95% accuracy on customer-provider edges. However, for the purposes of computing the worst congestion in the graph, this low accuracy of labeling is, in fact, enough. Indeed, as shown in Figure 7.1(b), labeling real graphs using our algorithm results in an error of less than 10% in the worst congestion

[4] False positives are nodes that are identified as stubs by the algorithm, but are not stubs in the ALR. False negatives are stubs in the ALR that are not identifies as stubs by the algorithm.

(while employing policy routing) in comparison with the congestion computed on ALRs. More importantly, the trends in congestion growth are identical in the two cases.

Other topologies. In addition to power law graphs, we also study congestion in power law trees and exponential topologies. A comparison of the former with power law graphs gives an insight into the significance of density of edges in the graph. The latter model is interesting because most generative models for power law topologies result in exponential distributions in the "fringe" cases [36]. Our power law tree topologies evolve according to the Preferential Connectivity model [21]. To generate exponential topologies, we modify Inet-3.0 to generate an exponential degree distribution first and then add edges in Inet's usual way. For a given n, the exponent β for the exponential graphs on n nodes is chosen such that the total number of edges in the exponential graph is very close to that of the corresponding power law graph on n nodes[5]. Note that due to a lack of real data for exponential graphs, we do not have a good way of labeling edges and nodes in them. Therefore, we do not perform experiments with policy routing or the leaf-2-leaf and clout traffic models for them.

7.2 Analytical Results

In this section, we show that the expected maximum edge congestion in a power law graph, specifically the congestion on the edge between the two highest degree nodes, grows as $\Omega(n^{1+\frac{1}{\alpha}})$ with n, when we route a unit flow between all pairs of vertices over the shortest path between them. We consider the Preferential Connectivity Generative Model of Barabasi et al. [21]. This model uses a fixed constant parameter k. The model starts with a complete graph on $k+1$ nodes. This set of nodes is called the *core* of the graph. The graphs grows in time-steps. Let the graph at time i be denoted G^i. At time step $i+1$, one node is added to the network. This node picks k nodes at random from G^i and connects to them. Each vertex v has a probability $\frac{d_v^i}{D^i}$ of getting picked, where d_v^i is the degree of the vertex at time i, and D^i is the total degree of all nodes at time i. At the end of n steps, with $k = 3$, this process is known to generate a power law degree distribution. Also, it is easy to see that in the resulting power law graph (for that matter in any power law graph), the maximum node degree is $n^{1/\alpha}$.

In order to show a lower bound on the congestion of a power law graph, our plan is roughly as follows. We consider the edge between the two highest degree nodes in the core—s_1 and s_2. Call this edge e^*. For every vertex v in the graph, we consider the shortest path tree T_v rooted at vertex v. We show that in expectation, $\Omega(n)$ such trees contain the edge e^*. Moreover, in these trees, the expected number of nodes in the subtree rooted at edge e^* is at least $\Omega(n^{\frac{1}{\alpha}})$. This gives us the lower bound in the following way: the routes taken by each connection are precisely those defined by the above shortest path trees; thus the congestion on any edge is the sum of congestions on the edge in these shortest path trees. Now, as described above, in $\Omega(n)$ shortest path trees, the congestion on

[5]We employ heuristic hill-climbing to estimate the value of the exponent β that minimizes error in the number of edges.

edge e^* is at least $\Omega(n^{\frac{1}{\alpha}})$. Therefore, the total congestion on edge e^* is at least $\Omega(n^{1+\frac{1}{\alpha}})$. Note that e^* is not necessarily the most congested edge in the graph, so the maximum congestion could be even worse than $\Omega(n^{1+\frac{1}{\alpha}})$. We get the following theorem:

Theorem 1 *The expected value of the maximum edge congestion in a power law graph with exponent α grows as $\Omega(n^{1+\frac{1}{\alpha}})$ with n, when we route a unit flow between all pairs of vertices over the shortest path between them.*

In the following, the distance between two nodes refers to the number of hops in the shortest path between the two nodes. We make a few technical assumptions. We assume that $1 < \alpha < 2$, and s_1 and s_2 are the highest degree nodes in the graph. For reasonably "small" numbers h, we assume that for any node v in the graph, the number of nodes within distance h of v is less than the number of nodes within distance h of s_1. In other words, s_1 is centrally placed in the graph. Here, "small" refers to distance around s_1 that contains lesser than half the nodes. These assumptions are justified by experimental evidence and some prior analysis [41] of the preferential connectivity generative model.

We begin with a technical lemma.

Lemma 1 *Let r be the maximum integer for which at least $\frac{n}{2}$ vertices lie at a distance $r+1$ or beyond from s_1. Then, $\Omega(n)$ nodes lie within distance $r-1$ of every node in the core of the graph. In particular, for any node in the core, $\Omega(n)$ nodes lie at a distance exactly $r-1$ from it.*

Proof: We prove that at least $\Omega(n)$ nodes lie within a distance $r-2$ of s_1. Then, since all vertices in the core are neighbors of s_1, these $\Omega(n)$ nodes lie within a distance $r-1$ of any vertex in the core of the graph. We begin by showing that at least $\Omega(n)$ nodes lie within a distance r of s_1, and then extend this to nodes at distance $r-1$ and $r-2$. Let level i denote the set of nodes at distance exactly i from s_1.

Remove from the graph all vertices that are at level $r+2$ or higher. The remaining graph has at least $\frac{n}{2}$ vertices, by the definition of r. Now, assume that there are at least $\frac{n}{10}$ vertices at level $r+1$, otherwise, we already have $> \frac{2n}{5}$ nodes in levels 0 through r, implying that $\Omega(n)$ nodes lie within distance r of s_1.

Now, let the number of nodes at level r be x. All the nodes in level $r+1$ in the residual graph are connected to nodes in level r. So, their number is at most the size of the neighbor set of level r. Now, in the best possible case, the nodes in level r could be the highest degree nodes in the graph. In this case, the minimum degree of any node in level r is given by y with $ny^{-\alpha} = x$. We get $y = \left(\frac{n}{x}\right)^{\frac{1}{\alpha}}$.

Now, the size of the neighborhood of level r is at most the total degree of nodes in the level. This is given by

$$\int_y^{n^{\frac{1}{\alpha}}} z\alpha n z^{-\alpha-1} dz \quad = \quad \frac{\alpha n}{\alpha - 1}\left(y^{1-\alpha} - n^{\frac{1}{\alpha}-1}\right)$$

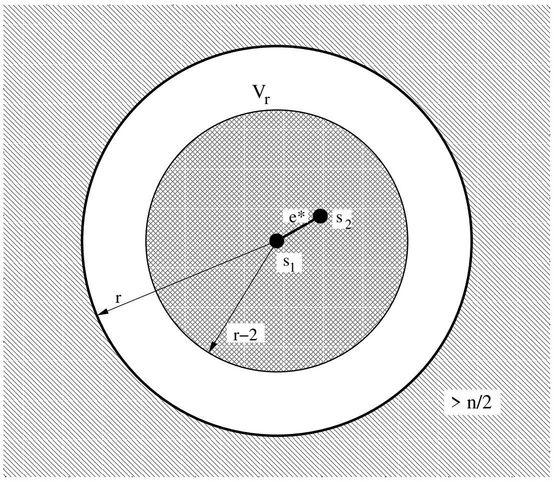


Figure 7.2: The model: A pictorial view of the graph and the set V_r

$$= \frac{\alpha n^{\frac{1}{\alpha}}}{\alpha - 1}(x^{1-\frac{1}{\alpha}} - 1)$$

This quantity is at least $\frac{n}{10}$ by our assumption above. Thus we get that $x = \beta n$, where $\beta = \left(\frac{1}{10}(1 - \frac{1}{\alpha})\right)^{\frac{\alpha}{\alpha-1}}$. This is a constant fraction of n.

Now, we can apply the same technique to compute the number of nodes at level $r - 1$ and then, $r - 2$. We get that the number of nodes at level $r - 2$ is at least $\left(\beta^{\alpha}(1 - \frac{1}{\alpha})\right)^{\frac{\alpha}{(\alpha-1)^2}} n$, with β as given above. ■

Let r be the distance defined by the above lemma. Let V_r denote the set of nodes that are within distance $r - 1$ of every node in the core of the graph (see Figure 7.2). By lemma 1, we have $|V_r| = \Omega(n)$. Now, the proof of the theorem has two parts. The first shows that many trees T_v corresponding to $v \in V_r$ contain the edge e^*.

Lemma 2 *The expected number of shortest path trees T_v, corresponding to nodes $v \in V_r$, that contain the edge e^* is $\Omega(n)$.*

Proof: Consider the tree T_v for some node $v \in V_r$. This is essentially a breadth first tree starting from v. If s_1 and s_2 are at the same level in the tree, then the edge e^* is not contained in the tree. On the other hand, if the nodes are at different depths in this tree, let s_1 be closer to v without loss of generality. In this case, one shortest path from v to s_2 is via s_1 and since we break ties in favor of paths with high degree nodes, T_v will contain this path via s_1. This implies that e^* is contained in the tree. Thus, trees containing e^* correspond to those v that are not equidistant from s_1 and s_2. We now show that there are $\Omega(n)$ nodes $v \in V_r$ that are not equidistant from s_1 and s_2. This implies the result.

First, observe that if we pick a random node in the graph, then conditioned on the fact that this node lies at a distance $d-1$, d or $d+1$ from s_2, there is at most a constant probability that this node lies at distance d from s_2. This is because using an argument congruent to that in lemma 1, we can show that the number of nodes at distance $d-1$ from s_2 is a constant fraction of the number of nodes at distance d.

Now, consider the nodes at distance $r-2$ from s_1. These are at least $\Omega(n)$ in number (lemma 1) and lie in V_r. Given that a node v is picked from this set, v is at a distance $r-3$, $r-2$ or $r-1$ from s_2. By the above argument, the probability that this node lies at distance $r-2$ from s_2 is at most a constant. Thus $\Omega(n)$ nodes in this set are *not* at distance $r-2$ from s_2 and we are done. ■

Next we prove that in any tree T_v ($v \in V_r$) containing e^*, e^* has a high congestion.

Lemma 3 *Let T_v be a shortest path tree, corresponding to $v \in V_r$, that contains the edge e^*. Then the expected congestion on edge e^* in this tree is $\Omega(n^{1/\alpha})$.*

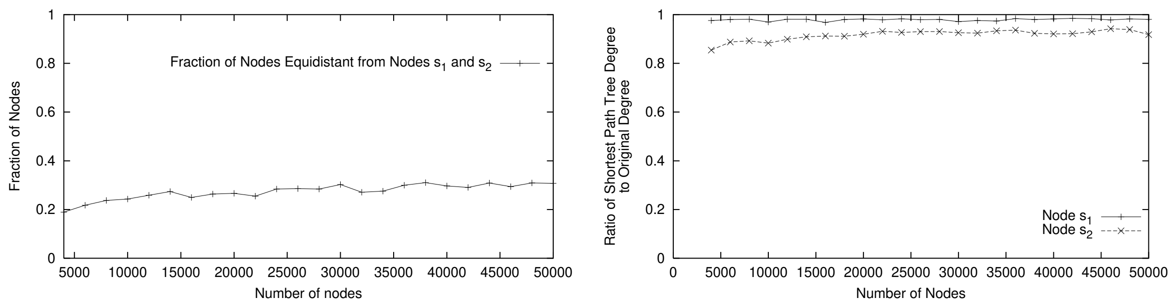
Proof: Without loss of generality, let s_1 be closer to v than s_2. We show that the degree of s_2 in T_v is $\Omega(n^{1/\alpha})$. This implies the result. Let level i denote the set of nodes at distance i from v in the tree.

Let d be the distance between v and s_2. All neighbors of s_2 lie in levels $\geq d-1$ in the tree. Note that $d \leq r-1$. Therefore by our assumption, the number of nodes lying at levels $\geq d+1$ in the tree is at least the number of nodes at distance r or greater from s_1. This number is at least $\frac{n}{2}$, by the definition of r. Let W denote the set of nodes that lie at levels $\geq d-1$ in the tree, and that arrived in the graph after step $\frac{n}{4}$. Note that there are at least $\frac{n}{4}$ nodes at level $d+1$ or higher that are in set W. Therefore, a constant fraction of the nodes in W lie at levels $\geq d+1$ in the tree.

First observe that the probability that a node entering the graph at time step t attaches to s_2 is roughly $t^{\frac{1}{\alpha}-1}$. This probability increases as the graph becomes larger and larger, as this is related. By removing the first quarter of the nodes entering the graph from consideration, and using the fact that these nodes are less likely to attach to s_2 than nodes arriving later, we conclude that the number of neighbors of s_2 that arrive after step $n/4$ is at least $3/4$th of the total degree of s_2.

Now all neighbors of s_2 lie at levels $\geq d-1$ in the tree. Then, by the observation in the previous paragraph, we have that at least $3/4$th of the neighbors of s_2 lie in the set W. Note that when a node in W entered the graph, the size of the graph varied between $\frac{n}{4}$ and n nodes. The probability that this node attached to s_2 varied between $n^{\frac{1}{\alpha}-1}$ and $(\frac{n}{4})^{\frac{1}{\alpha}-1} < 4n^{\frac{1}{\alpha}-1}$. Thus each node in W is roughly equally likely to attach to s_2 (within a factor of 4).

Now the degree of s_2 in the tree is at least the number of its neighbors in W that lie at levels $\geq d+1$. Using the fact that a constant fraction of the nodes in W lie at levels $\geq d+1$ in the tree, we get that a constant fraction of the neighbors of s_2 lie at levels $\geq d+1$ in the tree, in expectation. The result follows from the fact that the degree of s_2 is $\Omega(n^{\frac{1}{\alpha}})$. ■



(a) (b)

Figure 7.3: (a) Simulation support for the analytical model: Figure (a) shows the fraction of shortest path trees that do not contain the edge e^*. Figure (b) plots the ratio of degrees of s_1 and s_2 in a random shortest path tree to their degrees in the graph.

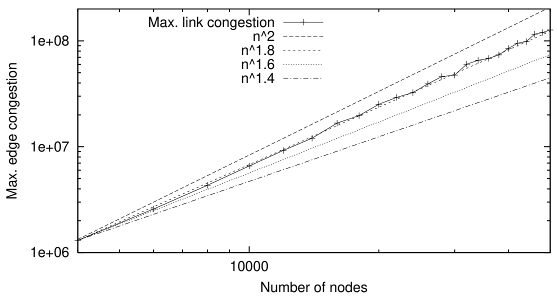
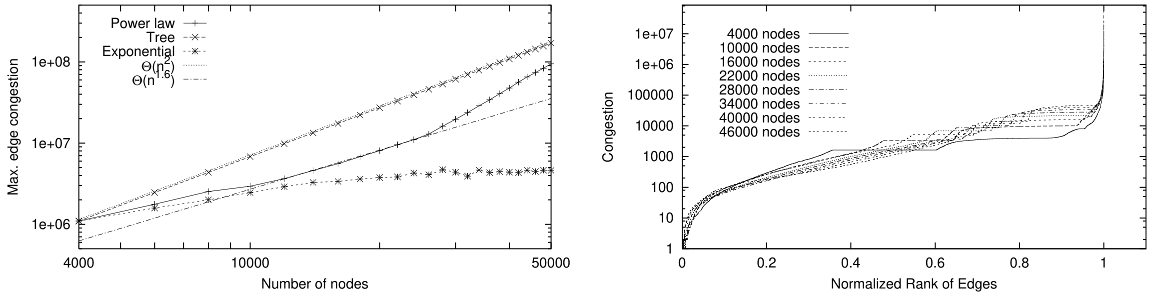


Figure 7.4: Maximum edge congestion: Plotted as a function of n, in Inet-3.0 generated graphs, with $\alpha = 1.23$. The figure also plots four other functions to aid comparison – $n^{1.4}$, $n^{1.6}$, $n^{1.8}$, n^2.

7.2.1 Experimental Support

In this section, we report experimental results to show that the theoretical results obtained above hold not just for the Preferential Connectivity Model, but also for Internet-like AS graphs generated by Inet-3.0. Unfortunately, the graphs generated by Inet-3.0, have different values of α for different n. This is consistent with the observed properties of the Internet's AS graph, that α decreases with time. (We discuss this in further detail in the subsequent section). In order to validate our theoretical results and observe the asymptotic behavior of congestion for a fixed value of α, we modify the Inet-3.0 code so that it always uses a fixed value of $\alpha = 1.23$, instead of recalculating it for every value of n. Each reported value is an average over multiple runs of the simulation, corresponding to different random seeds used for generating the graphs.

Figure 7.3(a) plots the fraction of nodes that are equidistant from s_1 and s_2. Note that this fraction always lies below 0.4 and is consistent with our result in Lemma 2 that at least a constant fraction of the trees, $\frac{n}{2}$ in this case, contain the edge e^*. Figure 7.3(b) compares the degrees of the two highest degree nodes in the graph to their corresponding degrees in the shortest path tree



(a) Maximum edge congestion

(b) Distribution of edge congestion

Figure 7.5: Edge congestion with shortest path routing and any-2-any communication: The figure on the left shows the maximum edge congestion. The figure on the right shows the distribution of congestion over all links, with the number of links normalized to 1 in each case. The figure on the left also plots the worst congestion for exponential graphs and preferential connectivity trees.

corresponding to some random node v. We find that the ratio of the two degrees for s_1 is consistently above 0.9. Similarly, the ratio of the two degrees for s_2 is always above 0.8 and increasing. This is consistent with the findings of Lemma 3.

Finally, we plot the maximum congestion in graphs generated by Inet-3.0, as a function of the number of nodes in the graph, in Figure 7.4. Note that the maximum congestion scales roughly as $n^{1.8}$, which is exactly $n^{1+1/\alpha}$ for $\alpha = 1.23$. This corroborates our finding in Theorem 1.

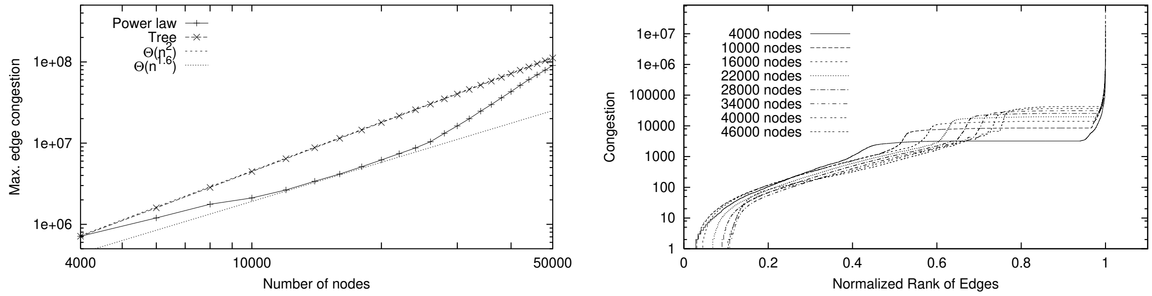
7.3 Simulation Results

In this section, we present the results from our simulation study over Inet-generated graphs. Henceforth, we shall use the graphs generated by Inet 3.0 *as is*, that is, we do not alter the way Inet chooses α to depend on n. In what follows, we first show results for shortest-path routing, followed by policy-based routing. In both cases, we first present results for the any-2-any communication model, then for the leaf-2-leaf model and finally for the clout model.

7.3.1 Shortest-Path Routing

Figure 7.5(a) shows the maximum congestion in power law graphs generated by Inet-3.0 as a function of the number of nodes. We use the any-2-any model of communication here. From the trend in the graph, it is clear that the maximum congestion in Internet-like power law graphs scales as $n^{1+\Omega(1)}$ or worse. Notice also that the slope of the maximum congestion curve is slightly increasing. This can be explained as follows. As mentioned earlier, Inet-3.0 chooses the exponent of the power law degree distribution as a function of the number of nodes n: $\alpha = at + b$, where $t = \frac{1}{s} log \frac{n}{n_0}$, $a = -0.00324$, $b = 1.223$, $s = 0.0281$ and $n_0 = 3037$[6]. Notice that the absolute value

[6] a, b and s are empirically determined constants. n_0 is the number of ASes in the Internet in November 1997.



(a) Maximum edge congestion

(b) Distribution of edge congestion

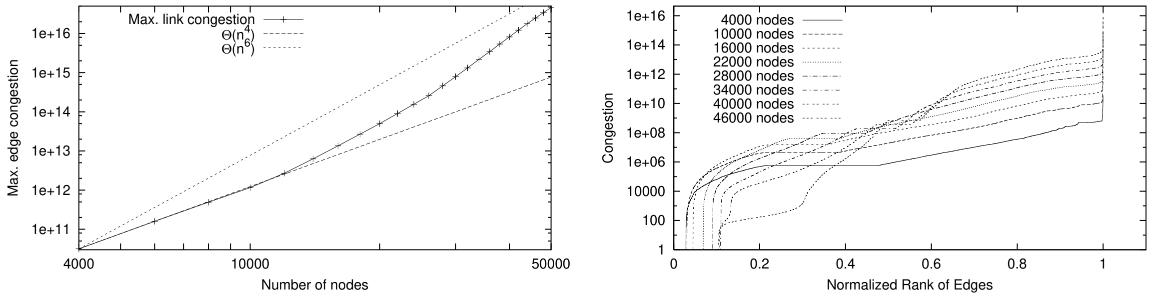
Figure 7.6: Edge congestion with shortest path routing and leaf-2-leaf communication: The figure on the left shows the maximum edge congestion. The figure on the right shows the distribution of congestion over all links (again normalized).

of α decreases as n increases, and so, as our lower bound of $\Omega(n^{1+1/\alpha})$ suggests, the *slope* of the function on a log-log plot should steadily increase. In fact around $n = 28000$, α becomes less than 1 and at this point we expect the curve to scale roughly as $O(n^2)$, which is the worst possible rate of growth of congestion.

The figure also shows the maximum congestion in power law trees and exponential graphs. The power law trees we generate, have the exponent α between 1.66 and 1.8, the value increasing with the number of nodes in the tree. These exponents are significantly higher than those of the corresponding power law graphs. Notice that the edge congestion on power law trees grows much faster compared to graphs which is expected since trees have much fewer edges. Our lower bound on the maximum congestion, which holds equally well for trees satisfying power law degree distributions, predicts the slope of the curve for trees to be at least 1.5, which is consistent with the above graph.

On the other hand, we notice that edge congestion in exponential graphs is much smaller compared to power law graphs. In fact, edge congestion in exponential graphs has an approximately linear growth. This could be explained intuitively as follows: Recall that for each n, we choose the exponent β of the exponential distribution so as to match the total number of edges of the corresponding n-node power law graph. Because the power law distribution has a heavier tail compared to the exponential distribution, the latter has more edges incident on low degree nodes. Consequently, low degree vertices in an exponential graph are better connected to other low degree vertices. Edges incident on low degree nodes “absorb” a large amount of congestion leading to lower congestion on edges incident on high degree nodes. As n increases the degree distribution becomes more and more even, resulting in a very slow increase in congestion.

In Figure 7.5(b), we show the congestion across all links in a power law graph. Notice that at higher numbers of nodes, the distribution of congestion becomes more and more uneven. The corresponding set of graphs for the leaf-2-leaf communication model is shown in Figure 7.6. The worst congestion is consistently about 0.8 times the worst congestion for the any-2-any model (not explic-



(a) Maximum edge congestion

(b) Distribution of edge congestion

Figure 7.7: Edge congestion with shortest path routing and clout model of communication: The figure on the left shows the maximum edge congestion. The figure on the right shows the distribution of congestion over all links (again normalized).

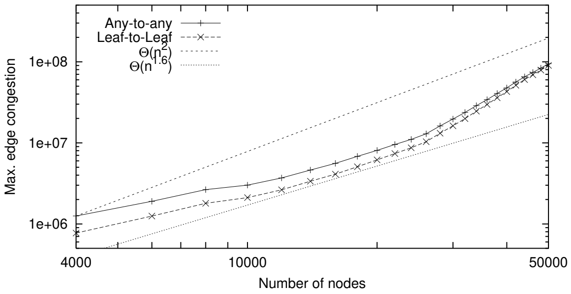
itly shown in the graph). The congestion across all edges, plotted in Figure 7.6(b), also displays a similar trend as for the any-2-any model. The distribution becomes more uneven as the number of nodes increases.

The results for the clout model are more interesting with the resulting maximum congestion in the graph scaling much worse than before. Indeed, as Figure 7.7(a) shows, the maximum congestion scales worse than n^5. This is because the total traffic in the graph also grows roughly as $O(n^4)$. Again, as with the any-2-any model, the smaller absolute values of α in the graphs generated by Inet-3.0 for larger values of n is the reason for the increasing slope of the curve. The graph of the congestion across all edges in this model, shown in Figure 7.7(b), is equally interesting. Compared to Figure 7.6(b) of the leaf-2-leaf model, Figure 7.7(b) looks very different: the unevenness in congestion is much more pronounced in the clout model of communication. In other words, the non-uniform traffic demand distribution only seems to exacerbate the already poor congestion scaling of the Internet-like graphs.

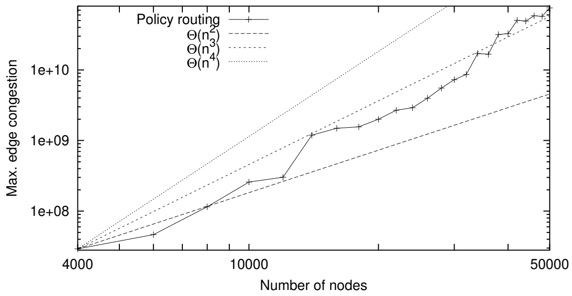
7.3.2 Policy-Based Routing

Figure 7.8 shows the maximum edge congestion for the three communication models, when policy based routing is used. For the any-2-any and leaf-2-leaf models, shown in Figure 7.8(a), the maximum edge congestion scales almost identically to that for shortest path routing (compared with Figure 7.5(a) and 7.6(a)). However, somewhat surprisingly, for the clout model (Figure 7.8(b)), congestion under policy based routing scales only as n^3 compared to over n^5 for shortest-path routing.

Figure 7.9(a) compares maximum congestion obtained for policy routing to that for shortest path routing. Notice that the two curves are almost overlapping, although policy routing seems to be slightly worse when the graph is small and gets better as the graph grows larger. This observation can be explained as follows: policy routing disallows certain paths from being used and could, in general, force connections to be routed over longer paths. This would increase the overall traffic in

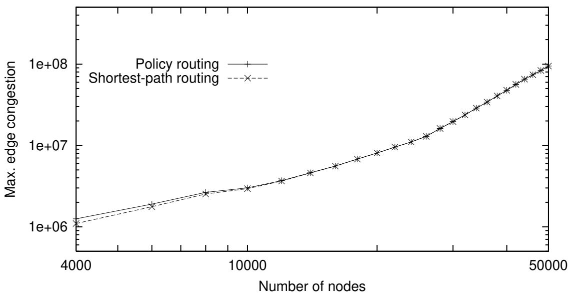


(a) Any-2-any and Leaf-2-leaf communication

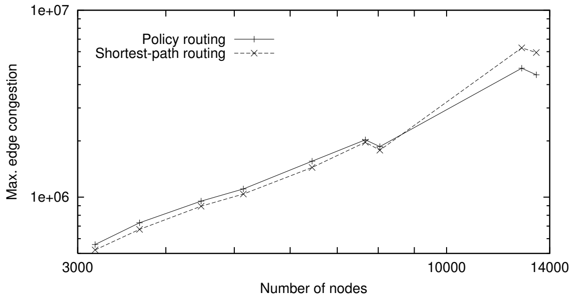


(b) Clout model

Figure 7.8: Policy-based routing: Maximum Edge congestion with policy-based routing in HLSs.



(a) Edge congestion on synthetic graphs



(b) Edge congestion on real graphs

Figure 7.9: Policy-based vs shortest path routing: Comparison of edge congestion for shortest path and policy based routing in the any-2-any model

the network leading to higher congestion, especially for a smaller graph size. However, as the size of the graph grows, there are more and more shortest paths available. As a result, the constraints placed by policy-based routing might not have any significant impact on the path lengths in the graph. In fact, at higher numbers of nodes, policy routing could provide better congestion properties, albeit only marginally different, than shortest path routing. This is because while shortest path routing always picks paths that go over high degree nodes, a fraction of these paths might not be allowed by policy routing as they could involve more than one peering edge. In this case, policy routing moves traffic away from the hot-spots, thereby, partially alleviating the problem. We believe that this is also the reason for the congestion scaling in the clout model to be better when considering policy-routing, as opposed to shortest-path routing.

In order to verify that the above observation is not just an artifact of our machine learning-based labeling algorithms, we plot the same curves for ALRs in Figure 7.9(b). These display exactly the same trend—policy routing starts out being worse than shortest path, but gets marginally better as n increases. To summarize, policy routing does not worsen the congestion in Internet-like graphs, contrary to what common intuition might suggest. In fact, policy routing might perform marginally

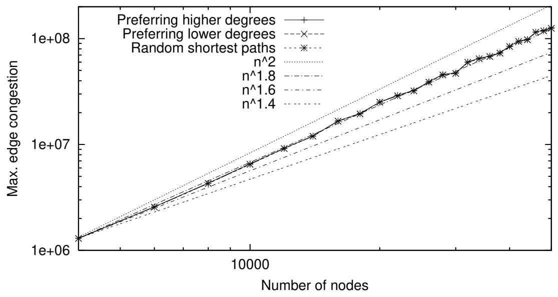


Figure 7.10: Tie-breaking rules in shortest-path routing: $\alpha = 1.23$. The figure plots the three different variations of breaking ties in shortest path routing.

better than shortest path routing.

7.3.3 Shortest Path Routing Variations

As mentioned in Section 7.1.1, in the shortest path routing scheme, whenever there are multiple shortest paths between two nodes, we pick the path that contains higher degree nodes to route the flow between them. It may appear that the poor congestion properties of power law graphs are a result of this tie breaking rule, and an alternate rule that favors low degree nodes may perform better by alleviating the congestion on high degree nodes. In order to confirm that our results are robust across various tie-breaking rules, we performed the experiments with two variants of the tie-breaking rule: favoring paths that contain lower degree nodes, and choosing a random shortest path when there is a choice of more than one.

For these experiments, we set α to be a constant value of 1.23 in Inet 3.0 and compare the resulting relations between maximum edge congestion and the number of nodes. As Figure 7.10 depicts, there is no noticeable difference between the three types of tie-breaking methods. The same holds true for leaf-2-leaf and clout models of traffic (results are omitted for brevity). This is because very few vertex pairs have multiple shortest paths between them. We thus conclude that our scheme of breaking ties by favoring paths containing higher degree nodes does not skew our results.

7.4 Improving the Congestion Scaling Properties

Our analytical and simulation results have shown that the maximum congestion in Internet-like power law graphs scales rather poorly in the graph size—$\Omega(n^{1+\Omega(1)})$. Our results show that edges between high degree nodes, which are typically peering edges between backbone carriers in the Internet core, are likely to get congested more quickly over time than other edges. In such a situation, to enhance the scaling properties of the network, it might become necessary to either change the routing algorithm employed by ASes in the Internet (i.e., BGP-style routing) or alter the intercon-

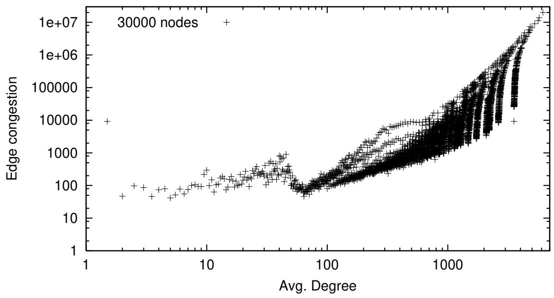


Figure 7.11: Degree vs congestion: Edge Congestion versus the average degree of the nodes incident on the edge (any-2-any model with shortest path routing). The congestion is higher on edges with a high average degree.

nection. Next, we address the latter issue of altering the structure of the Internet graph. Specifically, we focus on mechanisms for increasing the parallelism in edges between neighboring nodes in the Internet AS-graph.

7.4.1 Adding Parallel Network Links

We examine ways in which additional links can be placed in the network, so as to contain the effect of bad scaling of the maximum congestion. Specifically, we consider the model in which each link can be replaced by multiple links (between the same pair of nodes) that can share the traffic load[7]. Ideally, we would like to provide sufficient parallel links between a pair of nodes, so that the total congestion on the corresponding edge divided equally among these parallel links, even in the worst case, grows at about the same rate as the size of the network. The number of parallel links between a pair of nodes may need to change as the network grows to achieve this goal. Notice that this change does alter the degree-structure of the graph, but the alteration is only due to increased connectivity between already adjacent nodes[8]. In other words, this does not require new edges between nodes that were not adjacent before.

In fact, the network, at an AS level, already incorporates this concept of parallel links. For example, the power law structure of the AS graph only considers the adjacency of ASes: the link between Sprint and AT&T, for instance, is modeled by a single edge. However, in the real world the Sprint and AT&T ASes are connected to each other in a large number of places around the world. However, not much is known about the degree of such connectivity in the Internet today.

In order to guide the addition of parallel edges between adjacent nodes, we first observe that there is clear correlation between the average degree and edge congestion. Figure 7.11 plots the congestion of each edge against the average degree of the nodes on which it is incident. We show the results for shortest path routing on an Inet generated graph of 30000 nodes where any-2-any

[7]For results on alternate methods of alleviating congestion, please refer to [6].

[8]Note that the routing is still done based on the original degrees of nodes.

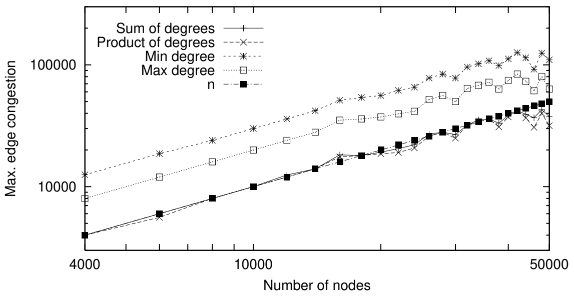


Figure 7.12: Alleviating congestion: Maximum relative congestion for shortest path routing, any-2-any model, when parallel links are added to the graph using the sum, product and max functions.

communication is used. The figure shows that edges incident on high degree nodes have much higher congestion than those incident on lower degree nodes. This suggests that a good choice for the number of parallel links substituting any edge in the graph, could depend on the degrees of nodes which an edge connects.

We examine several ways of adding parallel links based on the above observation. In particular, we let the number of links between two nodes be some function of the degrees of the two nodes and we consider the following functions: (1) sum of degrees of the two nodes, (2) product of the degrees of the two nodes, (3) maximum of the two degrees and, (4) minimum of the two degrees. We then compute the maximum relative congestion for these functions, that is, the maximum over all edges, of the congestion on the edge divided by the number of parallel links corresponding to each edge. The maximum congestion on Internet graphs for these models of adding new edges is shown in Figure 7.12. Notice that, surprisingly, when parallel links are added according to any of the above four functions the maximum relative congestion in the graph scales linearly. This implies that adding parallelism in the edges of Internet-like graphs according to the above simple functions is enough to ensure that uniform scaling of link capacities (for example, based on Moore's Law-like technology trends) can maintain uniform levels of congestion in the network and avoid persistent hot-spots.

7.5 On Networking Modeling and Congestion Scaling

There have been several theoretical studies aimed at studying the properties of large-scale, Internet-like graphs, as well as those analyzing congestion scaling properties of general graphs. In this section, we present a brief overview of some of these studies.

Of studies aiming to characterize properties of Internet-like graphs, one class has proposed various models of graph evolution that result in a power law degree distribution. Notable examples include the power law random graph model of Aiello et. al. [1], the bi-criteria optimization model of Fabrikant et. al. [36] and the Preferential Connectivity model of Barabasi and Albert [21, 12].

Another class of studies in this category [37, 86, 109] is aimed at analyzing the properties of power law graphs. However, most of these are based on inferences drawn from measurements of real data. The primary application of this latter class of studies is to construct realistic generators [73, 115, 109] for Internet-like graphs.

The problem of characterizing congestion in graphs, and specifically designing routing schemes that minimize congestion, has been studied widely in approximation and online algorithms. The worst congestion in a graph is inversely related to the maximum concurrent flow that can be achieved in the graph while obeying unit edge capacities. The latter is, in turn, related to a quantity called the cut ratio of the graph. Aumann et. al. [17] characterize the relationship between maximum concurrent flow and cut ratio: The maximum concurrent flow that can be achieved in a graph is always within a factor of $O(\log n)$ of the cut ratio, where n is the number of nodes. Okamura et. al. [82] give bounds on the cut ratio for special graphs. Algorithmic approaches to the problem (see [64, 65] for a survey) use a multi-commodity flow relaxation of the problem to find a fractional routing with good congestion properties (wherein demand between pairs of nodes is split across multiple paths). Although fairly good approximation factors have been achieved for the problem, most of the proposed routing schemes are not distributed, involve a lot of book-keeping, or require solving large linear programs, which makes them impractical for wide-area Internet routing.

Perhaps the work that shares similar goals as ours is that of Gksanditis et al. [44]. Using arguments from max-flow min-cut theory, their paper shows that graphs obeying power law degree distribution have good expansion properties in that, they allow routing with $O(n \log^2 n)$ congestion, which is close to the optimal value of $O(n \log n)$ achieved by regular expanders. In a follow-up paper, Mihail et al. [74] prove similar results on the expansion properties of graphs generated using the Preferential Connectivity model. The results presented in this chapter, in contrast with these two contemporary papers, focus specifically on commonly-used routing algorithms such as policy routing and shortest path routing. We show that power law graphs exhibit poor scaling properties with respect to these routing algorithms.

7.6 Analysis Caveats, Summary of Observation and their Implications

In this chapter, we addressed the question of how the worst congestion in Internet-like graphs (specifically at the AS-level) scales with the graph size. The key observations are shown in Table 7.1. Using a combination of analytical arguments and simulation experiments, we showed that maximum congestion scales poorly in Internet-like power law graphs. Our simulation results showed that the non-uniform demand distribution between nodes only exacerbates the congestion scaling. However, we found, surprisingly, that policy routing between adjacent ASes may not worsen the congestion scaling on power law graphs and might, in fact, be marginally better when compared to shortest-path routing.

We note that with the current trend of the growth of the Internet it is possible that some locations

The expected maximum congestion in Internet-like Preferential Attachment power law graphs with unit traffic demands and shortest path routing scales as $n^{1+\Omega(1)}$, where n is the number of nodes in the graph.
When non-uniform traffic matrices are considered, the congestion scaling properties worsen significantly.
Policy routing results in similar, if not marginally better, congestion than shortest path-based routing.
The poor congestion scaling properties of the Internet graph can be fixed using very simple heuristics to alter the topology of the network. For example, adding parallel edges between adjacent nodes in proportion to the minimum of their degrees can result in a linear scaling of congestion.

Table 7.1: Congestion Scaling in the Internet: Summary of key observations regarding the scaling properties of the Internet.

in the network might eventually become perpetual hot-spots. Fortunately, however, there is an intuitively simple fix to this problem. Adding parallel links between adjacent nodes (ASes) in the graph according to simple functions of their degrees will help the maximum congestion in the graph scale linearly. In this case, it might not be necessary for the capacity of some links in the graph to grow at a faster rate than the others. In fact, a natural evolution of link capacities according to Moore's Law may be sufficient to accommodate the growing traffic demand in the network.

Next, we discuss important caveats in our analysis and simulations.

Analysis Caveats. We would like to mention that the results presented in this chapter may not hold in general for all power law graphs. Our results (both simulation-based and analytical) are meant for graphs representing Internet connectivity at the AS level. These results, may not apply to power law random graphs [1]. Note also that while the preferential connectivity model is known to yield graphs with a similar degree distribution as the AS-level graph, it is not clear whether the model accurately captures the AS-level connectivity dynamics (e.g., economic considerations for peering). That said, our simulations on measured AS-level graphs show that our key observations hold for the existing AS graph. Therefore, if the current dynamics of connectivity between ASes continues to hold in the future, we can expect our results to hold for future AS-level graphs too.

Our analysis does not extend to router-level graphs. Analyzing the router-level topology is much harder compared to the AS-level graph due to three reasons: (1) Not much is known about the topology of Internet's router-level graph. Most existing maps of the Internet's router-level topology (such as Rocketfuel maps [105, 28]) are considered incomplete; (2) IP-level routing cannot be modeled easily using shortest path routing or simple inter-domain policy-based routing, since this would require knowledge of traffic engineering employed by ASes in the Internet; (3) Finally, some researchers have used power law graphs resulting from probabilistic models (such as power law random graph models [1]) to approximate the router-level connectivity (see for example [109]). However, recent work has shown that such models are error-prone since they do not explicitly consider the technological and economic constraints or trade-offs behind router interconnections [66]. Graphs arising from such trade-offs are referred to as *Heuristically Optimal Topologies*. However, there are no analytically-tractable models for generating such topologies. A thorough analysis of the router-level interconnection is a challenging open problem.

The key results from our study of congestion scaling in the Internet graph may be simply summarized as follows: The congestion along edges in the Internet graph is likely to scale poorly with the growing size of the network. As a result, end-networks cannot employ routing-based mechanisms such as route control to extract good performance from the future network. However, simple heuristics for adding parallel edges between adjacent vertices in the graph can help significantly improve the congestion scaling properties.

Chapter 8

Conclusions and Open Problems

In this chapter, we outline the contributions of this thesis and present questions for future consideration.

8.1 Thesis Summary

This dissertation sought to answer the following central question: *What routing-based mechanisms can well-connected end-networks employ to improve their Internet access experience?* Specifically, we were interested in understanding if end-networks today required special support from the Internet routing protocol suite or the routing infrastructure to obtain better performance. Our answer to this question, in short, is *No*. It is sufficient for end-networks to simply make clever use of a small number of routes per destination, as determined by the Internet's routing protocol. To achieve better performance in this manner, end-networks do not require changes to the current routing protocol, or support from special-purpose infrastructures, but can instead rely on purely end point-based mechanisms.

8.2 Contributions

This dissertation makes several important contributions. The foremost contributions are the analysis of the benefits of multihoming route control and the implementation and evaluation of a simple, practical route control system. In addition, we present the first-ever characterization of performance bottlenecks inside ISP networks. We also study the impact of the growth of congestion at these bottlenecks on future end-to-end performance. We present an overview of these contributions next.

8.2.1 Properties of wide-area bottlenecks

A key challenge in optimizing the Internet performance of well-connected end-networks is to identify the heavily-loaded network links that constrain performance. For years, common knowledge of wide-area bottleneck links was limited to folklore—these bottlenecks were widely believed to be confined to the edges of Internet domains (e.g., peering links), where link utilization is supposedly high.

In this dissertation, we conducted the first quantitative study of the characteristics of typical wide-area bottleneck links. We probed a large number of paths between well-connected machines located at universities and routers inside various carrier ISPs. We developed a suite of tools to automatically identify and characterize bottleneck links along these paths. Using these tools, we found that bottleneck links with very low available bandwidth are prevalent in the wide-area Internet, especially inside or between small regional providers. We also deduced significant correlations between the likelihood of wide-area links appearing as bottlenecks and the latencies of the links. Finally, we observed that, contrary to popular perception, wide-area bottlenecks are almost evenly split between peering and intra-domain links.

8.2.2 Benefits of multihoming route control and comparison with overlay routing

Although constrained bottlenecks exist in the wide-area Internet, the Internet's rich topology makes it possible to "route around" them. Past studies for circumventing wide-area performance bottlenecks advocated using *overlay routing*. In this approach, end-points can route their traffic via intermediate "overlay" nodes deployed around the Internet. This helps end-points bypass the default routes determined by Internet's routing protocol (BGP), and avoid bottlenecks along these routes.

In contrast, we believed that a much simpler approach called *Multihoming Route Control* can offer similar performance improvements as overlay routing. In this strategy, an end-network buys connectivity from a few different ISPs and intelligently schedules its traffic across the ISPs. The idea of multihoming route control was introduced a few years ago by commercial products such as RouteScience and SockEye. However, little was known about the true extent of the benefits of these products.

In this dissertation, we quantified the potential benefits of multihoming route control in improving the Web download performance of Internet end-points, as well as their resilience to service interruptions. Using Internet-scale experiments conducted in collaboration with Akamai Technologies, we showed that by multihoming to three ISPs, and intelligently scheduling transfers across the ISPs, an end-network could potentially improve its Internet response times, transfer speeds and availability by up to 30%, relative to using a single ISP connection. We also showed that employing more than 3 ISP connections offers little additional benefit, but the ISPs themselves must be carefully selected in order to realize the maximum possible improvements.

Since multihoming route control is BGP routing-compliant, it cannot provide nearly the same route selection flexibility as overlay routing. Using Internet-scale measurement, we showed that despite this seemingly limited flexibility, multihoming route control offers only *marginally* inferior Internet performance than overlay routing. For example, the transfer speeds from multihoming to three ISPs are at most 10% inferior relative to overlay routing. This observation suggests that there is definite hope of extracting good performance from the BGP protocol, and therefore, it is unnecessary to replace BGP by a different protocol altogether.

8.2.3 Route control in practice

We extended the above work on the potential benefits of route control by implementing and evaluating a multihomed route control system. We showed that, in practice, route control products could employ very simple design and operational principles to extract nearly-optimal Internet performance from multihoming. Using trace-driven emulations of an enterprise network with 3 ISP connections, we showed that our proposed schemes could improve the Web performance of multihomed end-networks by about 25% when compared to using a single ISP. Furthermore, the Web performance from our proposals is at most 10% away from the best possible performance. We also exposed important flaws in the design of current route control appliances. For example, we showed that the conventional practice of employing historical measurement samples to monitor and predict ISP performance could, in fact, result in sub-optimal performance.

8.2.4 Congestion scaling at bottlenecks

The observations regarding multihoming showed that today's routing protocols and topology can support good Internet performance. Over time, the Internet will grow in size and traffic volumes will increase. At the same time, ISPs will upgrade network link capacities to accommodate the growing traffic load. In this thesis, we observed that despite the improvement of link speeds in the future, the Internet's topology and routing may, in fact, cause the load on certain links in the network to increase at a much faster rate than on others. Such links could soon evolve into *persistent bottlenecks*, and, in turn, significantly limit future Internet performance. Specifically, we analyzed a simple model of the routing and topology of the Internet at an Autonomous System level (AS-level), and showed that the congestion at key links in the network may grow as poorly as $n^{1+\Omega(1)}$, where n is the number of ASes. This result implies that we may have to carefully alter the Internet's topology and/or routing to guarantee good end-to-end performance in the future network. To this end, using large-scale simulations, we showed that small fixes to the Internet's AS-level interconnections could drastically reduce the congestion in the network. For example, we observed that adding parallel edges between adjacent ASes in proportion to the minimum of their degrees can ensure good Internet performance.

8.3 Lessons for the Longer Term

Our measurement of the benefits of multihoming route control established the efficacy of the technique in the current Internet. At the same time, we believe that certain key observations from our study of multihoming will continue to be applicable for future Internet architectures. For example, this dissertation argues for richer first-hop connectivity at network end-points. Irrespective of how we redesign the Internet of the future, we believe that this basic principle will always offer several advantages to endpoints, such as the freedom to route traffic via the ISP of their choice and the ability to optimize for specific performance metrics without relying on ISPs to provide the necessary knobs. Our work also shows that end-to-end performance optimization does not have to be coupled with optimizing the behavior routing protocol itself. Therefore, research on the latter problem can progress in parallel with the development of tools for endpoint route control. Another lesson we learn is that while it would be nice for the Internet's routing protocol to be performance aware, all we really need, at least from the perspective of optimizing end-to-end performance, is a protocol that can provide us with a reasonable choice of routes to select from. The current BGP protocol is quite effective at supporting such an interface for choosing routes. Of course, BGP suffers from other pathologies, like lack of security, unpredictability and slow reconvergence, which need to be addressed separately in our quest for a better wide-area routing protocol for the future.

8.4 Future Work

In what follows, we discuss key issues left open by this dissertation. Where applicable, we outline possible approaches to address them.

8.4.1 Longer-term Measurement Analyses

In our study of wide-area Internet bottlenecks, we relied on measurements collected over short time-scales to derive the likelihood of links of various types appearing as bottlenecks. The most natural extension is to investigate the "persistence" of bottlenecks: On what time-scales does congestion on wide-area bottlenecks change? Similarly, our observations regarding the benefits of route control are based on data collected over a week-long period. It is useful to complement these observations with analyses of changes in ISP performance over longer time-scales, e.g., 6 months or 1 year. This would also help us understand if the choice of the best set of ISPs for multihoming is likely to change over long time periods, and if so, what the impact on the subscriber performance (and subscription cost) may be.

8.4.2 Explaining Diminishing Returns

In Section 4.3, we showed that there is little benefit from multihoming to more than three ISPs. Informally, this can be explained by the fact that a fourth ISP can add only a limited amount of diversity to that already provided by three well-chosen ISPs. We believe that the diminishing returns can be explained formally by analyzing the hierarchical nature of ISP interconnections. As explained in Section 3.1.3, the ISP hierarchy is composed of 5 tiers. Further, the highest tier consists of under 20 ISPs, with vast global reach. This implies that typical end-to-end paths traverse one or more tier-1 ISPs with a high likelihood. Another implication is that paths via distinct ISPs (say of tiers 2 or 3) to random Internet destinations are very likely to merge in a tier-1 network. The greater the degree of overlap (in terms of the number of routers or ISPs), the lesser the benefit of employing multiple ISPs. We believe that it is possible to model the likelihood for end-to-end paths to overlap with each other, and then, to extend this model to explain diminishing returns from multihoming.

8.4.3 Global Effects of Multihoming Route Control

We mentioned in Section 6.4 that Qiu et al. [45] study the interactions between multiple route control agents at equilibrium. The authors conclude that the impact of individual route control actions on global RTT performance is minimal, assuming an equilibrium exists. This work leaves several issues open for consideration. For example, can route controlled traffic attain an equilibrium at all? What are the dynamics of interactions between route control agents at, or when converging to, an equilibrium? Does widespread deployment of route control result in adverse interactions with ISP traffic engineering by making traffic load due to any single end-network vastly unpredictable?

Further, ISPs may alter their pricing structures in response to route control, in order to "smooth out" the traffic and/or to attract more customers. It is unclear if this diminishes the benefits from route control that we identified in this study. Finally, as route control is more widely adopted, it is open to debate if the best strategy for a new end-network is to stay connected to the best single ISP, or to itself adopt route control.

We would like to mention here that any potential negative effects of wide-spread deployment of route control, such as traffic fluctuations, could be mitigated, to a certain extent, by employing "third-party route control services", such as Internap Inc. [53]. In this model, a third-party providers servicing a large metropolitan area buys high-capacity uplinks to several large ISPs. End-customers buy a connectivity service from the third-party provider, whereby they hand their traffic over to the service provider which in turn selects on of its many uplinks to send it over. In making a choice of which ISP to send over, the service provider could optimize for several goals such as load-balancing across ISP uplinks and minimizing traffic fluctuations on any single ISP link.

From the perspective of ISPs, this approach is better than allowing individual route control at endpoints since it ensures that the multihomed traffic aggregate from a metropolitan area behaves in a more predictable, smooth fashion. Also, in comparison with overlay service providers, route control providers do not run the risk of violation of policies. As a result, it may both be easier to deploy such services as well as cheaper to subscribe to them in comparison with overlays.

At the same time, in comparison with endpoint control, the third-party route control approach limits the flexibility of endpoints. For example, endpoints can no longer optimize for specific performance goals, such as throughput as opposed to latency. Second, if endpoints were to coordinate ISP and route selection with their remote destinations (this is common if the two ends were branch offices of a single parent organization), then allowing a third party service provider to control routing may offer inferior performance than allowing endpoints full control.

8.4.4 New Changes to BGP

In this dissertations, we showed that BGP routes can support good performance in the Internet today. We also observed, in Section 5.2.7.2, that cooperative inter-domain traffic exchange between ISPs can further reduce the differences between route control and overlay routing. There are several approaches to encouraging cooperation between neighboring ISPs while ensuring that the ISPs do not reveal anything about their internal structure to their neighbors (see, for example, [68]). However, these approaches work only for pairs of neighboring ISPs. An interesting open issue is to understand whether these techniques can be extended to end-to-end paths, and what functional changes to BGP this might require. It is possible that such changes to BGP may completely eliminate the differences between overlay routing and BGP-based path selection.

8.4.5 Better Models for Congestion Scaling

Our analysis of the scaling of congestion in the Internet graph is limited to the interconnections between ISPs. It is unclear if the ISP interconnection graph is likely to evolve so as to maintain a power law structure. As large ISPs acquire and merge with other small or large ISPs, it is likely that, in future Internet, ISPs will belong to one of two kinds: large ISPs which own and operate backbones (such as current tier-1 ISPs), and very small ISPs (such as current tier-4 ISPs) catering to home or DSL connections. It would be interesting to study the scaling properties of these alternate interconnections.

Also, as we stated in Section 7.6, our analysis of congestion scaling does not extend to the Internet's router-level graph. We believe that it is possible to develop simple approximations to the router-level topology, and investigate congestion scaling using either analysis or simulations. Further, our analysis neglects the impact of technologies such as content distribution networks (CDNs). By serving data from locations close to end-clients, CDNs may contribute to alleviating traffic load from key non-access Internet links. This effect can be modeled in our analysis by forcing path lengths to be upper-bound by a small constant.

Bibliography

[1] W. Aiello, F. Chung, and L. Lu. Random evolution in massive graphs. In *Proc. of the 42th Annual Symposium on Foundations of Computer Science*, pages 510–519, 2001.

[2] Akamai Technologies: Content Distribution Service. `http://www.akamai.com`, 1999.

[3] Akamai Technologies: EdgeScape. `http://www.akamai.com/en/html/services/edgescape.html`, 2004.

[4] Akamai Technologies: SureRoute. `http://www.akamai.com/en/html/services/sureroute_for_failover.html`, June 2001.

[5] A. Akella, S. Chawla, A. Kannan, and S. Seshan. Scaling Properties of the Internet Graph. In *Proc. of the Twenty Second Annual ACM Symposium on Principles of Distributed Computing*, Boston, MA, July 2003.

[6] A. Akella, S. Chawla, A. Kannan, and S. Seshan. Scaling Properties of the Internet Graph. Technical Report CMU-CS-03-131, CMU, Pittsburgh, Pennsylvania, May 2003.

[7] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A Measurement-Based Analysis of Multihoming. In *Proc. of ACM SIGCOMM '03*, Karlsruhe, Germany, August 2003.

[8] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh. A Comparison of Overlay Routing and Multihoming Route Control. In *Proc. of ACM SIGCOMM '04*, Portland, OR, August 2004.

[9] A. Akella, S. Seshan, and A. Shaikh. An Empirical Evaluation of Wide-Area Internet Bottlenecks. In *Internet Measurement Conference*, Miami, FL, November 2003.

[10] A. Akella, S. Seshan, and A. Shaikh. Multihoming Performance Benefits: An Experimental Evaluation of Practical Enterprise Strategies. In *Proc. of the USENIX 2004 Annual Technical Conference*, Boston, MA, June 2004.

[11] A. Akella, S. Seshan, S. Shenker, and I. Stoica. Exploring Congestion Control. Technical Report CMU-CS-02-139, CMU CS, Pittsburgh, Pennsylvania, June 2002.

[12] R. Albert and A.-L. Barabasi. Topology of evolving networks: local events and universality. *Physical Review Letters*, 85(24):5234–5237, 2000.

[13] L. Amini, A. Shaikh, and H. Schulzrinne. Issues with Inferring Internet Topological Attributes. In *Proceedings of SPIE ITCOM*, August 2002.

[14] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris. Resilient Overlay Networks. In *Proc. of the 18th Symposium on Operating System Principles*, Banff, Canada, October 2001.

[15] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Rao. Improving Web Availability for Clients with MONET. In *Proc. Second Networked Systems Design and Implementation*, May 2005.

[16] D. Applegate and E. Cohen. Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs. In *Proceedings of ACM SIGCOMM*, Pittsburgh, PA, August 2003.

[17] Y. Aumann and Y. Rabani. An $O(\log k)$ Approximate Min-Cut Max-Flow Theorem and Approximation Algorithm. *SIAM Journal on Computing*, 27(1):291–301, 1998.

[18] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. *Overview and Principles of Internet Traffic Engineering*. Internet Engineering Task Force, May 2002. RFC 3272.

[19] B. Fortz and J. Rexford and M. Thorup. Traffic engineering with traditional IP routing protocols. In *IEEE Communications Magazine*, October 2002.

[20] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. In *Proceedings of 21st ACM Symposium on Principles of Database Systems (PODS 2002)*, Madison, WI, June 2002.

[21] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–6512, 1999.

[22] BBC News, UK Edition. Full speed ahead for Japan's broadband. http://news.bbc.co.uk/1/hi/technology/3278375.stm, 2003.

[23] BGP Tables from the University of Oregon RouteViews Project. `http://moat.nlanr.net/AS/data`.

[24] C. F. Bornstein. SureRoute Performance and Subscription, July 2003. Personal communication.

[25] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proc. of ACM SIGCOMM '94*, August 1994.

[26] T. Brecht, D. Pariag, and L. Gammo. accept()able Strategies for Improving Web Server Performance. In *Proc. of USENIX Annual Technical Conference*, June 2004.

[27] CAIDA: Cooperative Association for Internet Data Analysis. CAIDA Tools. `http://www.caida.org/tools/`, 2001.

[28] CAIDA Tools: Skitter. `http://www.caida.org/tools/measurement/skitter/`.

[29] N. Cardwell, S. Savage, and T. Anderson. Modeling TCP Latency. In *Proc. of IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[30] R. L. Carter and M. E. Crovella. Measuring bottleneck link speed in packet-switched networks. *Performance Evaluation*, 27–28:297–318, October 1996.

[31] T. H. Cormen, C. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 1998. Chapter 26.

[32] A. Downey. Using pathchar to Estimate Internet Link Characteristics. In *Proceedings of ACM SIGCOMM*, Cambridge, MA, August 1999.

[33] Emulab: Network Emulation Testbed. `http://www.emulab.net/`.

[34] C. Estan and G. Varghese. New directions in traffic measurement and accounting. In *Proc. of ACM SIGCOMM '02*, Pittsburgh, PA, August 2002.

[35] F5 Networks: BIG-IP Link Controller. `http://www.f5.com/f5products/bigip/LinkController/`.

[36] A. Fabrikant, E. Koutsoupias, and C. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the Internet. In *ICALP*, pages 110–122, 2002.

[37] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the Internet topology. *Proc. of ACM SIGCOMM '99*, August 1999.

[38] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing Iceberg Queries Efficiently. In *Internaational Conference on Very Large Databases (VLDB)*, New York, August 1998.

[39] N. Feamster, D. Andersen, H. Balakrishnan, and M. F. Kaashoek. Measuring the Effects of Internet Path Faults on Reactive Routing. In *Proc. of ACM SIGMETRICS 2003*, June 2003.

[40] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for Interdomain Traffic Engineering. *ACM SIGCOMM Computer Communication Review*, October 2003.

[41] T. Fenner, A. Flaxman, and A. Frieze. High Degree Vertices and Eigenvalues in the Preferential Attachment Gr aph. In *RANDOM 03*, pages 264–274, 2003.

[42] L. Gao. On Inferring Autonomous System Relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9(6), December 2001.

[43] L. Gao and F. Wang. The Extent of AS Path Inflation by Routing Policies. In *Proc. of IEEE GLOBECOM 2002*, pages 2180–2184, 2002.

[44] C. Gkantsidis, A. Saberi, and M. Mihail. Conductance and Congestion in Power Law Graphs. In *Proc. of ACM SIGMETRICS 2003*, 2003.

[45] D. Goldenberg, L. Qiu, H. Xie, Y. Yang, and Y. Zhang. Optimizing Cost and Performance for Multihoming. In *Proc. of ACM SIGCOMM '04*, Portland, OR, 2004.

[46] R. Govindan and V. Paxson. Estimating router ICMP generation delays. In *Proc. of Passive and Active Measurement Workshop (PAM)*, Fort Collins, CO, 2002.

[47] A. Gunnar, M. Johansson, and T. Telkamp. Traffic Matrix Estimation on a Large IP Backbone: A Comparison on Real Data. In *Second Internet Measurement Conference (IMC) 2004*, Taormina, Italy, October 2004.

[48] F. Guo, J. Chen, W. Li, and T. Chiueh. Experiences in Building a Multihoming Load Balancing System. In *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[49] J. Guojun, G. Yang, B. R. Crowley, and D. A. Agarwal. Network Characterization Service (NCS). In *Proc. of IEEE International Symposium on High Performance Distributed Computing (HPDC)*, San Francisco, CA, August 2001.

[50] N. Hu, L. Li, M. Mao, P. Steenkiste, and J. Wang. Locating Internet Bottlenecks: Algorithms, Measurements, and Implications. In *Proc. of ACM SIGCOMM '04*, Portland, OR, August 2004.

[51] Y. Hyun, A. Broido, and k claffy. Traceroute and BGP AS Path Incongruities. Technical report, CAIDA, University of California, San Diego, 2003. `http://www.caida.org/outreach/papers/2003/ASP/`.

[52] IETF Traffic Engineering Working Group. `http://www.ietf.org/html.charters/tewg-charter.html`, 2000.

[53] Internap Network Services: Flow Control Platform. `http://www.internap.com`.

[54] A. Iyengar and J. Challenger. Improving Web Server Performance by Caching Dynamic Data. In *USENIX Symposium on Internet Technologies and Systems*, 1997.

[55] V. Jacobson. pathchar – A Tool to Infer Characteristics of Internet Paths. ftp://ee.lbl.gov/pathchar/, 1997.

[56] M. Jain and C. Dovrolis. Pathload: A Measurement Tool for End-to-end Available Bandwidth. In *Proc. of Passive and Active Measurement Workshop (PAM)*, Fort Collins, CO, March 2002.

[57] Jeffrey Semke and Jamshid Mahdavi and Matthew Mathis. Automatic TCP Buffer Tuning. *ACM SIGCOMM Computer Communication Review*, 28(4), October 1998.

[58] A. Khanna and J. Zinky. The Revised ARPANET Routing Metric. In *Proc. of ACM SIGCOMM '89*, pages 45–56, Austin, TX, September 1989.

[59] E. Koutsoupias and C. Papadimitriou. Worst-Case Equilibria. *Lecture Notes in Computer Science*, 1563:404–413, 1999.

[60] B. Krishnamurthy and J. Wang. On network-aware clustering of Web clients. In *Proceedings of ACM SIGCOMM*, Stockholm, Sweden, August 2000.

[61] B. Krishnamurthy and C. Wills. Improving Web Performance by Client Characterization Driven Server Adaptation. In *Proc. of the International World Wide Web Conference*, 2002.

[62] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, June 2001.

[63] K. Lai and M. Baker. Nettimer: A Tool for Measuring Bottleneck Link Bandwidth. In *Proc. of USENIX Symposium on Internet Technologies and Systems*, March 2001.

[64] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufmann Publishers, 1992.

[65] S. Leonardi. On-line Network Routing. In *Online Algorithms - The State of the Art*, pages 242–267. Springer, 1998.

[66] L. Li, D. Alderson, W. Willinger, and J. Doyle. A First-Principles Approach to Understanding the Internet's Router-Level Topology. In *Proc. of ACM SIGCOMM '04*, 2004.

[67] B. A. Mah. *pchar*: A Tool for Measuring Internet Path Characteristics. `http://www.employees.org/~bmah/Software/pchar/`, June 2000.

[68] R. Mahajan, D. Wetherall, and T. Anderson. Negotiation-Based Routing Between Neighboring ISPs. In *Proc. Second Networked Systems Design and Implementation*, May 2005.

[69] Z. Mao, J. Rexford, J. Wang, and R. Katz. Towards an Accurate AS-Level Traceroute Tool. In *Proc. of ACM SIGCOMM '03*, Karlsruhe, Germany, August 2003.

[70] M. Mathis and J. Mahdavi. Diagnosing Internet Congestion with a Transport Layer Performance Tool . In *Proc. INET '96*, Montreal, Canada, June 1996.

[71] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgement options. Request for Comments 2018, Internet Engineering Task Force, October 1996.

[72] J. McQuillan, I. Richer, and E. Rosen. The New Routing Algorithm for the ARPANET. *IEEE Transactions on Communications*, 28(5):711–719, May 1980.

[73] A. Medina, A. Lakhina, I. Matta, and J. Byers. Brite: An approach to universal topology generation. In *MASCOTS*, 2001.

[74] M. Mihail, C. Papadimitriou, and A. Saberi. On Certain Connectivity Properties of the Internet Topology. In *FOCS 2003*, pages 28–35, 2003.

[75] T. M. Mitchell. *Machine Learning*. McGraw-Hill Companies, Inc., 1997.

[76] P. Morissey. Route optimizers: Mapping out the best route. *Network Computing*, December 2003. `http://www.nwc.com/showitem.jhtml?docid=1425f2`.

[77] E. M. Nahum, M. Rosu, S. Seshan, and J. Almeida. The effects of wide-area conditions on WWW server performance. In *Proc. of ACM SIGMETRICS*, Cambridge, MA, June 2001.

[78] National Laboratory for Applied Network Research. Routing data. http://moat.nlanr.net/Routing/rawdata/.

[79] Nortel Networks: Alteon Link Optimizer. `http://www.nortelnetworks.com/products/01/alteon/optimizer/`.

[80] W. B. Norton. Internet Service Providers and Peering. In *Proceedings of NANOG 19*, Albuquerque, NM, June 2000.

[81] ns-2: Network Simulator. http://www.isi.edu/nsnam/ns/, 2000.

[82] H. Okamura and P. Seymour. Multicommodity flows in planar graphs. *Journal of Combinatorial Theory*, B 31:75–81, 1981.

[83] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A Simple Model and its Empirical Validation. In *Proc. of ACM SIGCOMM '98*, pages 303–323, Vancouver, Canada, September 1998.

[84] J. Padhye and S. Floyd. Identifying the TCP Behavior of Web Servers. In *Proc. of ACM SIGCOMM '01*, August 2001.

[85] J. Pang, A. Akella, B. Krishnamurty, S. Seshan, and A. Shaikh. On the Responsiveness of DNS-Based Network Control. In *Second Internet Measurement Conference (IMC) 2004*, Taormina, Italy, October 2004.

[86] K. Park and H. Lee. On the effectiveness of route-based packet filtering for distributed dos attack prevention in power-law Internets. In *Proc. of ACM SIGCOMM '01*, San Diego, California, August 2001.

[87] V. Paxson. End-to-End Internet Packet Dynamics. *Proc. of ACM SIGCOMM '97*, pages 139–152, September 1997.

[88] PlanetLab. `http://www.planet-lab.org`, 2002.

[89] L. Qiu, Y. Yang, Y. Zhang, and S. Shenker. On Selfish Routing in Internet-Like Enviroments. In *Proc. of ACM SIGCOMM '03*, Karlsruhe, Germany, 2003.

[90] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure. Interdomain traffic engineering with BGP. In *IEEE Communications Magazine*, 2003.

[91] RADB whois Server. `whois.radb.net`.

[92] Radware: Peer Director. `http://www.radware.com/content/products/pd/`.

[93] Rainfinity: Overview of RainConnect. `http://www.rainfinity.com/products/wp_rc_overview.pdf`, 2004.

[94] Rether Networks: Internet Service Management Device. `http://rether.com/ISMD.htm`.

[95] RIPE whois Service. `whois.ripe.net`.

[96] M. Roughan, M. Thorup, and Y. Zhang. Traffic Engineering with Estimated Traffic Matrices. In *Internet Measurement Conference*, Miami, FL, November 2003.

[97] T. Roughgarden and E. Tardos. How Bad is Selfish Routing? In *IEEE Symposium on Foundations of Computer Science*, pages 93–102, 2000.

[98] RouteScience Technologies: PathControl. `http://www.routescience.com/products`.

[99] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The End-to-End Effects of Internet Path Selection. In *Proceedings of ACM SIGCOMM*, Boston, MA, September 1999.

[100] S. Savage et al. Detour: A Case for Informed Internet Routing and Transport. *IEEE Micro*, 19(1):50–59, 1999.

[101] J. Semke, J. Mahdavi, and M. Mathis. Automatic TCP Buffer Tuning. `http://www.psc.edu/networking/projects/auto/`.

[102] A. Shaikh, R. Tewari, and M. Agrawal. On the effectiveness of dns-based server selection. In *Proceedings of IEEE INFOCOM*, Anchorage, AK, April 2001.

[103] Sockeye Networks, Inc. `http://www.sockeye.com`.

[104] N. Spring, R. Mahajan, and T. Anderson. Quantifying the Causes of Internet Path Inflation. In *Proc. of ACM SIGCOMM '03*, August 2003.

[105] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *Proc. of ACM SIGCOMM '02*, Pittsburgh, PA, August 2002.

[106] J. W. Stewart. *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1999.

[107] Stonesoft: Multi-Link Technology. `http://www.stonesoft.com/files/products/StoneGate/SG_Multi-Link_Technology_Whitepaper.pdf`, October 2001.

[108] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz. Characterizing the Internet Hierarchy from Multiple Vantage Points. In *Proceedings of IEEE INFOCOM*, June 2002.

[109] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. Network Topology Generators: Degree-Based vs Structural. In *Proc. of ACM SIGCOMM '02*, Pittsburgh, Pennsylvania, August 2002.

[110] H. Tangmunarunkit, R. Govindan, and S. Shenker. Internet Path Inflation Due to Policy Routing. In *SPIE ITCOM*, August 2001.

[111] S. Tao, K. Xu, Y. Xu, T. Fei, L. Gao, R. Guerin, J. Kurose, D. Towsley, and Z.-L. Zhang. Exploring the Performance Benefits of End-to-End Path Switching. In *International Conference on Network Protocols*, November 2004.

[112] Tinyproxy. `http://tinyproxy.sourceforge.net`, November 2003.

[113] Traceroute.org: maintained by Thomas Kernen. `http://www.traceroute.org`.

[114] University of Oregon: RouteViews Project. `http://www.routeviews.org`.

[115] J. Winick and S. Jamin. Inet-3.0: Internet Topology Generator. Technical report, University of Michigan, 2002. Technical Report, CSE-TR-456-02.

[116] Yahoo! BB. `http://www.bbapply.com/`, 2005.

[117] X. Yang. NIRA: A New Internet Routing Architecture. In *Proc. of the ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA)*, August 2003.

[118] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the Constancy of Internet Path Properties. In *Proc. of ACM SIGCOMM Internet Measurement Workshop (IMW)*, November 2001.

[119] Y. Zhang, V. Paxson, and S. Shenker. The Stationarity of Internet Path Properties: Routing, Loss, and Throughput. Technical report, ICSI Center for Internet Research, May 2000.

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