Uncovering Performance Differences among Backbone ISPs with Netdiff

Ratul Mahajan Ming Zhang Lindsey Poole Vivek Pai
Microsoft Research *Princeton University*

Abstract – We design and implement Netdiff, a system that enables detailed performance comparisons among ISP networks. It helps customers and applications determine, for instance, which ISP offers the best performance for their specific workload. Netdiff is easy to deploy because it requires only a modest number of nodes and does not require active cooperation from ISPs. Realizing such a system, however, is challenging as we must aggressively reduce probing cost and ensure that the results are robust to measurement noise. We describe the techniques that Netdiff uses to address these challenges.

Netdiff has been measuring eighteen backbone ISPs since February 2007. Its techniques allow it to capture an accurate view of an ISP's performance in terms of latency within fifteen minutes. Using Netdiff, we find that the relative performance of ISPs depends on many factors, including the geographic properties of traffic and the popularity of destinations. Thus, the detailed comparison that Netdiff provides is important for identifying ISPs that perform well for a given workload.

1 Introduction

Knowledge of the performance characteristics of ISP networks is highly valuable. It can enable customers and applications to make informed choices regarding which ISP(s) to use for their traffic. These choices are important because the performance of distributed applications depends heavily on the network paths that are used.

Shedding light on ISP performance can also improve overall network infrastructure. Application performance in the Internet depends collectively on multiple ISPs. Unfortunately, the inability to differentiate individual ISPs' performance creates little incentive for ISPs to resolve problems and promote internal innovation [24]. In response, researchers have proposed radical network architectures based on ISP accountability, overlays or customer-directed routing [2, 5, 6, 24, 32, 41]. However, we believe that simply providing visibility into ISPs' performance creates the right incentives. For instance, no particular ISP is motivated to act if studies report that the average latency in the Internet is 60 ms. If instead, studies report that the average latency for the customers of an ISP is twice that for the customers of competitors, market forces will motivate the ISP to improve its performance.

It is thus surprising that the problem of systematically understanding how well various ISPs deliver traffic has received little attention, especially in the research community. To our knowledge, there has been only one commercial effort [20], whose limitations we discuss in the next section. Today, customers of ISP networks are often in the dark about which ISPs are better and if the higher price of a particular ISP is justified by better performance [25, 26, 35, 42]. A common method for customers to obtain this information is by asking each other about their experiences [25, 26, 42]. Similarly, distributed applications are unaware of how the choice of ISP impacts performance. Even if they use measurements to learn this [3, 14], they cannot predict the performance for ISPs to which they do not directly connect.

Motivated by the observations above, we consider the task of comparing the performance of ISP networks, both in the recent past and over longer time periods. We focus on large ISPs that form the backbone of the Internet. Collectively, these ISPs carry most of the application traffic in the Internet. Their customers include content providers, enterprises, universities, and smaller ISPs.

We first identify the important requirements for a system to compare ISPs. These requirements govern how the measurements should be conducted and analyzed. A key requirement is to quantify performance in a way that is relevant to customers and applications. This implies, for instance, that we measure the performance of paths that extend to destination networks, rather than stopping where the paths exit the ISP's network. The latter is common in service level agreements (SLAs) of ISPs today, but it is less useful because application performance depends on the performance of the entire path. Other requirements include enabling a fair comparison among ISPs, by taking into account the inherent differences in their sizes and geographic spreads, as well as helping ISPs improve their networks.

We then design and implement a system, called Netdiff. It is composed of a modest number of measurement nodes placed inside edge networks and does not require active cooperation from the ISPs themselves. It is thus easy to deploy. There are several challenges in making such a system practical, however. For instance, we must aggressively control probing overhead, which can be prohibitive if implemented naively; we devise a novel set covering-packing based method to systematically eliminate redundant probes. The body of the paper discusses these challenges in detail and describes how we address them. Our current implementation measures ISP performance in terms of path latency, which is a basic measure of interest for most applications. We are currently extending Netdiff to other relevant measures.

Netdiff has been operational on PlanetLab since February 2007. It currently measures eighteen backbone ISPs. We find that its methods are highly effective. For instance, it reduces the probing overhead by a factor of 400 compared to a naive probing technique. We also find that application performance is closely correlated with its inferences. An informal case study involving a real customer confirms that Netdiff is useful for ISPs' customers.

To further demonstrate the usefulness of Netdiff, we use it to compare the performance of the ISPs that it measures. We find that traffic performance can vary significantly with the choice of ISP, but no single ISP is best suited for all types of workloads. For instance, some ISPs are better at delivering traffic internationally, while others are better for domestic traffic; and some ISPs are better for traffic originating in certain cities, while others are better for certain other cities. We also find that the performance of paths internal to an ISP, which form the basis of typical SLAs, do not reflect end-to-end performance. Thus, selecting an ISP is a complex decision, and the detailed comparison enabled by Netdiff can be very helpful.

This paper considers only the technical aspects of ISP performance comparison and ignores other aspects such as acceptability to ISPs. Ultimately, the success of our effort depends on non-technical aspects as well and we have started investigating them. Encouragingly, in other domains, vendors have accepted performance comparison and even actively participate in the process [36, 39].

2 Related Work

In this section, we place our work in the context of existing methods to compare ISPs and measure networks.

2.1 Methods to Compare ISPs Today

There are two main options available today to customers who want to compare ISPs' performance.

Service-level agreements (SLAs) Many ISPs offer an SLA that specifies the performance that customers can expect. These SLAs are typically not end-to-end and specify performance only within the ISP's network. For instance, an SLA may promise that 95% of traffic will not experience latency of more than 100 ms inside the ISP's network. A few providers also offer "off-net" SLAs in which performance is specified across two networks – the ISP's own network and that of some of its neighbors.

Today's SLAs have two shortcomings when using them to compare ISPs. First, application performance depends on the entire path to the destination and not just on a particular subpath. As such, ISPs with better SLAs may not necessarily offer better performance (Section 7.2). Second, because they are independently offered by each ISP, SLAs make comparisons among ISPs difficult. Some SLAs may mention latency, some may mention loss rates, some may mention available capacity, and yet others may mention a combination of metrics. Even with comparable measures, further difficulties in comparison arise due to differences in the size and geographic spread of each ISP. For instance, is a 100-ms performance bound for an ISP with an international network better or worse than a 50-ms bound for an ISP with a nation-wide network? Our work uses measures that can be used to compare ISPs regardless of differences in their networks.

Third-party systems There exist systems that allow multihomed customers to select the best ISP for their traffic [3, 14]. These systems, however, enable comparison only among ISPs from which the customer already buys service. Our goal is to let customers compare arbitrary ISPs to guide their purchasing decisions in the first place.

There exist a few listings to compare arbitrary ISPs, but most of these are focused on broadband and dial-up ISPs [12, 17]. For backbone ISP comparison, which is our focus, we are aware of only Keynote [20]. It measures latency and loss rate for paths internal to ISPs and paths between pairs of ISPs. For these measurements, it co-locates nodes within some ISPs' points of presence (PoPs) and measures the paths between the nodes.

Keynote's approach for comparing ISPs has several limitations. First, because it requires active cooperation from ISPs to place nodes inside their PoPs, Keynote's coverage of an ISP's network is poor (Section 6.2). This node placement strategy is also more vulnerable to ISPs that wish to game the measurements as it is easier to separate probe and actual traffic. Second, like SLAs, it does not characterize the end-to-end path. Third, Keynote's probes are addressed to its measurement nodes and not actual destinations. Because Internet routing is destination-based, the performance experienced by destination-bound traffic may differ from measurement-node-bound traffic.

2.2 Other Work on Network Measurement

We draw heavily on works that infer ISP topologies [15, 19, 37]. We use many existing techniques, e.g., DNS-based mapping of IP addresses to geographical locations. We also extend some existing techniques. For instance, our set covering-packing abstraction for reducing probe traffic is a more general and flexible formulation than the heuristics used in Rocketfuel [37].

We also draw on systems that construct network-wide information sources to estimate the performance of various paths [13, 27, 31]. We share with them the challenge of controlling probing overhead. However, existing systems cannot be used for ISP comparison because they reduce overhead by abstracting away details crucial for such comparison. For instance, iPlane [27] views the network as a collection of links; to estimate performance between two hosts, it firsts estimates the series of links along the path and then bases path performance on link performances. In this procedure, errors in path estimates lead to incorrect performance estimates and the impact of destination-based routing is ignored. We believe that capturing these effects accurately is necessary for reliable ISP comparison, which led us to develop different methods for reducing probing overhead.

Our key contributions, however, lie not in developing new topology or performance measurement techniques but in formulating the problem of ISP comparison and building a practical system for it. Previously, ISP topology inference work has not measured performance and performance measurement work has not compared ISPs.

3 Goals and Approach

Our goal is to characterize how well traffic handed to an ISP is delivered to the destination. This traffic includes what is sent by the ISP's customers to various destinations and what is sent by the ISP's neighbors to the customers. Rather than performing a coarse characterization that ranks ISPs without regard to the properties of the traffic, we want to enable consumers to compare and decide which ISP is better for their specific traffic. Thus, we must uncover detailed differences in ISPs' performance along dimensions of interest to consumers. These include geographical location of end points, types of sources and destinations (e.g., content provider versus end users), and stability of performance. For instance, a content provider in Los Angeles should be able to determine which ISP delivers best performance to users in East Asia.

As the measure of performance, this paper focuses on latency, i.e., the time packets take to reach their destination after they are handed to the ISP. Latency has a first order impact on the performance of many applications (Section 6.4). Ongoing work is extending our system to other measures such as loss rate and available bandwidth.

3.1 Requirements for a System to Compare ISPs

Inspired by benchmarks for file systems and databases [9, 10, 40], we require our system to quantify performance in a way that is relevant to consumers, enable fair comparison, and help ISPs make informed optimizations. We describe these requirements in more detail below.

1. Relevant to applications and ISPs' customers The primary requirement is that our inferences be directly relevant to consumers; this has three implications:

i) Measure the performance of the entire path from where the traffic enters the ISP to the destination, not just the internal component. One might argue that the external component be discounted because the ISP does not control it directly. We argue for its inclusion because the performance of the entire path is what matters to applications. Additionally, the ISPs can influence the quality of the entire path, through routing and peering decisions.

ii) Reflect the experience of application traffic. This means that we use traffic addressed to destinations of interest and not extrapolate application performance from the performance of the underlying links. The latter may not reflect application experience because of possible routing issues. The desire to reflect application experience also suggests that we passively measure the performance of real traffic, but we defer this to future work.

iii) Along with a long-term, average view, capture performance over short intervals. Short-term views provide their own utility because they enable wide-area applications to make short-term adjustments, inform customers of the variance in an ISP's performance, and provide information on how an ISP performs during periods of increased importance to the customers (e.g., day versus night). Based on the timescales of routing dynamics in the Internet [22], we target a period of 15 minutes to capture a snapshot of an ISP's performance. As discussed later, this places a significant demand on our system.

2. Fair comparison across ISPs Our measures of ISP performance should account for inherent differences in ISP networks, such as their size and geographic presence. For instance, it is unfair to compare the average time that traffic spends in two ISP networks when one is international and the other is regional.

To account for differences among ISPs, instead of viewing them as networks of routers and links, we view them as networks that connect cities by inferring the locations of their routers. Combined with inferences about the geographical location of destination networks, it lets us normalize results based on the geographical distance between the end points. It also enables customers to focus exclusively on ISPs that serve their needs. For instance, some customers may be interested only in paths between Los Angeles and Europe.

3. Helpful to ISPs Our system should also help ISPs better understand their own performance. They should be able to tell, for instance, whether performance issues that customers experience stem from problems inside their own network or outside their network, and whether performance is particularly bad from certain cities. The resolutions of these problems are different in each case.

3.2 Architecture of Netdiff

Building a system to compare ISPs is challenging because ISPs are embedded in an inter-network of many other ISPs. Unlike file and database systems, we cannot bring an ISP network to a laboratory and construct a measurement harness around it [9, 10, 40]. Instead the

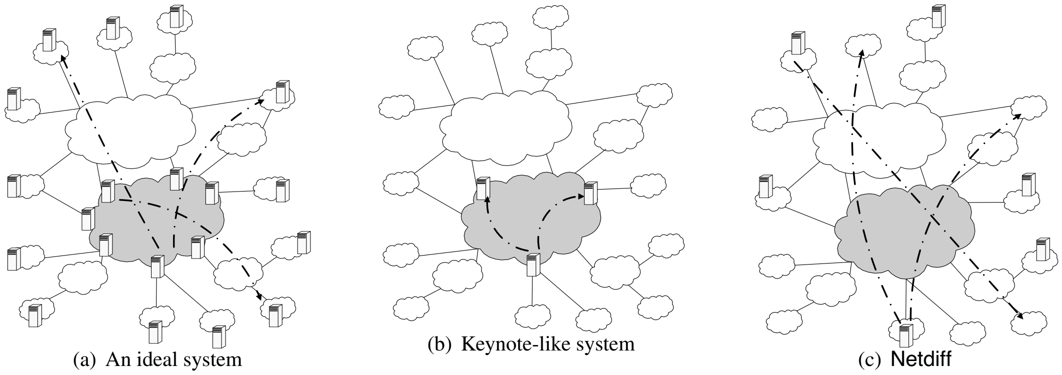


Figure 1: *Three different architectures for measuring ISP performance. The shaded cloud represents the target ISP network, and the boxes represent measurement nodes.*

ISP network must be measured *in situ*, but deploying such a harness can be difficult.

To understand how one might build a system to characterize an ISP network, consider an ideal system of Figure 1(a). This system has measurement nodes inside each PoP and each destination network because we want to measure performance from PoPs where the ISP is handed the traffic to destinations. With this system, the measurement task is straightforward but the deployment barrier is high. It requires thousands of measurements nodes to measure paths to even a fraction of destinations networks. It also requires significant cooperation from ISPs to place nodes inside their PoPs. Many ISPs may be unwilling or unable to provide this level of cooperation.

Keynote circumvents these problems by placing nodes inside only a few PoPs of cooperative ISPs and measuring only paths between those nodes. Figure 1(b) shows our understanding of Keynote's system. It, however, has the previously mentioned limitations: inability to measure paths to destinations and poor coverage (Section 6.2).

Netdiff, illustrated in Figure 1(c), approximates the capability of the ideal system but is easy to deploy. We place measurement nodes inside edge networks. To provide good coverage with a modest number of nodes, we use single-ended, traceroute-like probes to destinations. From these probes, we infer the performance of the relevant subpaths from entry into the ISP to the destination. We identify subpaths by first inferring the ISP's topology.

Another advantage of Netdiff over Keynote is robustness to ISPs that wish to game the measurements. It is harder to separate probe and actual traffic from hosts inside edge networks. Masking techniques can further raise the bar for this separation [30].

4 Measurement and Analysis Techniques

While our architecture is easy to deploy, engineering it is challenging. We must: i) aggressively reduce the number of probes; ii) extract performance information about the subpath of interest from the end-to-end probes; and iii) make our inferences robust to measurement noise. We consider each challenge in successive subsections.

4.1 Reducing Probing Requirement

Because we place measurement nodes inside edge networks, we must limit probing overhead to control the bandwidth cost for host networks and to not overload their access links. To understand the need for limiting the probing requirement, assume that there are 200 measurement nodes and 250 K destination IP prefixes, the current size of the BGP routing table. Also assume that there are twenty hops in a path and it takes a 100-byte probe to measure to each hop. Then, if we want to measure an ISP within 15 minutes probing from all nodes to all prefixes requires a prohibitive 1 Gbps of probing traffic. We use the following methods to reduce this overhead.

Use BGP atoms Instead of using IP prefixes as destinations, we use BGP atoms. Atoms are groups of prefixes whose paths and routing dynamics are similar [1, 8]. It is not necessary that *all* atoms, as inferred using BGP tables, are "atomic." But using atoms instead of prefixes significantly reduces the number of destinations and presents a worthwhile trade-off [27].

Select probes based on recent routing history Probes from a measurement node to many destinations do not traverse the ISP of interest and thus are not useful for measuring that ISP. We use a view of routing paths from the measurement node to restrict probing to paths that traverse the ISP. Before a node is used for measurements, it collects its view of routing to all the destinations. After that, this view is continuously refreshed using low-rate background probing.

Eliminate redundancies The set of all possible probes include many redundancies. For example, if probes from two nodes enter the ISP at the same place, only one is required. Similarly, for paths internal to an ISP, a probe that traverses three PoPs provides information about performance between three pairs of cities; other probes that traverse individual pairs are redundant. Eliminating such redundancies can lower the probing overhead and also balance load across nodes.

The redundancy elimination problem is the following. Suppose we know the path of each possible probe. Of these, we want to select a subset such that: i) each ISP city to destination network is probed; and ii) each internal path between two cities is probed; iii) probing load at a measurement node does not exceed a threshold.

The problem above is an instance of the set covering-packing problem [21]: given multiple sets over a universe of elements, pick a subset of input sets such that each element is included at least a given number of times (covering constraint) and no element is included more than a given number of times (packing constraint). In our case, the input sets are probes, and the elements are paths to destinations, internal paths, and measurement nodes. Probes typically contain all three element types. This formulation of redundancy elimination is more general than the three heuristics used in Rocketfuel [37]. It is also more flexible in that it can systematically assign load based on a node's ability.

The set covering-packing problem is NP-hard, but greedy heuristics are known to yield good solutions [21]. After casting our input data into a set covering-packing problem, we implement one such greedy heuristic. Probes are added to the measurement set until all elements are covered at least once. At each step, the probe that covers the most as yet uncovered elements is added.

4.2 Recovering Network Topology

To extract paths of interest, we need to discover where an ISP begins and ends in the measured path and map its IP addresses to their respective locations. We use existing methods based on DNS names and majority voting for this task [37, 43]. For instance, the name sl-gw12-sea-4-0-0.sprintlink.net corresponds to a router belonging to Sprint in Seattle. We extend *undns* [37] with new naming rules to increase the number of names that are successfully deciphered.

We infer the location of destination networks using the commercial geolocation database from MaxMind [29]. This database is compiled from websites that ask users for their location. MaxMind claims that its accuracy is 99% in determining the country of an IP address. Within a country its stated accuracy varies and is stated to be 80% within the USA. Its predictions are used only if they pass our tests (see below).

4.3 Dealing with Errors and Noise in Data

There are several sources of noise and errors in our data. To make our inferences robust, we design tests to detect sources of erroneous data and filter them appropriately.

IP to ISP mapping An IP address may be incorrectly mapped to an ISP, e.g., due to an erroneous DNS name. We check for such errors by observing the gathered traceroute data. The IPs belonging to the same ISP must appear consecutively, e.g., they should not be separated by an IP that belongs to another ISP. Transient routing problems can cause such an anomaly as well, but if we observe such an anomaly across many traceroutes, we can conclude that the ISP of the intervening IP has been incorrectly assigned.

Router IP to location mapping An IP address may be assigned an incorrect location, e.g., again due to an erroneous DNS name. We check for such errors using two tests. First, the traceroute for an ISP should not exit and then re-enter a city. As before, some of these anomalies arise because of transient routing issues; however, persistent issues indicate incorrect location mapping. The location mapping of an IP that is frequently sandwiched between two IPs belonging to a single different location is likely incorrect.

Second, we run a speed of light test among neighbors to detect erroneous mappings. The differences in the round trip latency observed to neighboring IPs should be more than the minimum time it takes for light to travel between the locations of the IPs. The latter time is calculated using the geographical coordinates assigned to particular locations and the speed of light in fiber. Thus, this test detects problems in assignment of geographic coordinates as well. If an IP fails this test for a majority of its neighbors, we conclude its location to be incorrect. Because of asymmetric routing, this test may fail for individual pairs even when the underlying data is correct.

Depending on the ISP, only 0.2-1.1% of traceroutes fail one of the two tests above. Deleting the mapping of a handful of IPs resolves the anomalies.

Geolocation for destination networks To detect errors in the geolocation database, we again use a test based on speed of light. Using traceroutes to the destination, we compare: i) the difference in the round trip latency between the destination and intermediate IPs with known locations; and ii) the minimum time for light to travel that path. Destinations for which the former is often less than the latter are deemed as having incorrect locations.

Path asymmetry Because we infer path latency using single-ended measurements, we must guard against our inferences being confused by significant asymmetries in forward and reverse paths. We discard traceroutes for which forward and reverse path length to an IP of interest differs by more than three. The reverse path length is inferred using the remaining TTL in the probe response. In Section 6.3, we show that this technique allows us to obtain reasonably accurate latency estimates by filtering out significantly asymmetries.

Overloaded node or local links Finally, in initial testing we found that our latency estimates were being corrupted by overloaded probing nodes or overloaded links

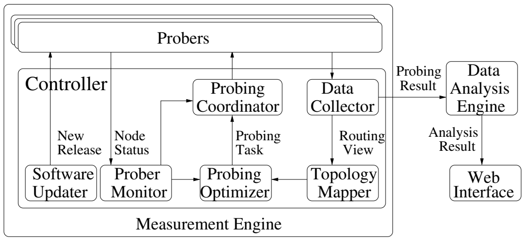


Figure 2: *Main functional modules of* Netdiff.

before the path entered the ISP. We now detect these events by observing the variance in the round trip time from the node to where the probe enters the ISP and discard data from nodes with high variance. A threshold of 40 ms worked well in our experiments. We demonstrate the effectiveness of this technique in Section 6.3.

5 Implementation of Netdiff

The implementation of Netdiff consists of measurement and data analysis engines and a Web interface, as illustrated in Figure 2. We describe each component below.

5.1 Measurement Engine

Netdiff divides the measurement process into cycles, measures one ISP per cycle, and iterates through the list of ISPs. This functionality is spread across a centralized *controller* and multiple *probers*. At the start of a cycle, the controller sends a list of probe destinations to each prober. When the probers finish, the probing results are recovered and the next cycle begins.

The key challenge in executing the process above is to start new cycles frequently, which is limited by two factors. The first one is the time for which the system not probing and is instead conducting some other support task. Done naively, even the simple task of transferring a single file to probers can run into tens of minutes. The second factor is the need to synchronize all probers at the beginning of each cycle because probers take different amounts of time to finish probing in a cycle. We control these factors through aggressive parallelization and termination of slow tasks.

Probers Probers measure path quality and maintain a fresh routing view to all BGP atoms. Upon receiving the list of destinations for a cycle, probers measure the paths to those destinations at the maximum allowed rate. Probing for the routing view is done at a low rate and spread over a day due to the large number ($\sim$55K) of atoms.

We use a customized version of traceroute for probing. Probes are constructed to maximize the chance that all probe packets for a destination take the same router-level path in the network [7]. We probe multiple hops of a path in parallel and probe multiple destinations in parallel, subject to the maximum rate of 60 Kbps.

Controller The controller has six submodules:

1. Topology mapper recovers ISP topologies from routing views (Section 4.2). It evolves inferred ISP topologies with time by expiring routers and links that have not been observed for a week. It also refreshes DNS names of IP addresses once a week. To enable analysis of old data, we save the current view of ISP topologies once a day.

We continuously monitor the quality of the current routing and topological views. The monitoring script looks for indicators such as the number of IP addresses in an ISP, the number of IP addresses not resolved by *undns*, and the number of anomalous traceroutes. This script guides if any action, e.g., adding new *undns* rules, is needed. It also brings major topological changes, such as ISP mergers, to our attention.

2. Probing optimizer generates a list of probe destinations for each prober, based on the target ISP, the list of currently live probers, and routing views. It uses the redundancy elimination algorithm described earlier.

3. Probing coordinator drives the measurement cycles. At the beginning of each cycle, it transfers a list of probe destinations to each prober. To guard against slow transfers, it uses a customized version of parallel *scp* which ensures that each file is either completely copied within a minute or not copied at all. When it finds from the prober monitor (below) that all probers have finished, it moves to the next cycle. To prevent a measurement cycle from being blocked by a slow prober, it terminates probing activity after 15 minutes. Depending on the target ISP, 88-100% of the probers are able to finish their tasks in time. If not finished before, after this time limit, all probers are ready for the next cycle. Probers can usually measure roughly 9K paths per cycle.

4. Prober monitor periodically polls each prober to determine its current status – dead, busy, or idle.

5. Data collector copies the routing view and probing results from the probers to the controller.

6. Software updater ensures that all probers have up-to-date probing software and configuration files.

The controller performs almost all of the tasks in parallel. In our earlier implementation that performed many tasks sequentially, we found that much time was spent not probing because any delay (e.g., in data collection) slowed the entire chain. In our current version, the only tasks that are not performed in parallel with probing are transferring destination lists and checking prober status towards the end of a cycle. These tasks take roughly 2 minutes, which lets us start a new cycle every 17 minutes, of which 15 are spent probing.

5.2 Data Analysis Engine

Netdiff converts raw measurements into results that can be used to compare ISPs in three steps. The first step is to extract information about the paths of interest, i.e., those between pairs of cities of an ISP and between an ISP's city and a destination atom. Consider an example path, observed while measuring AT&T's network:

```
Traceroute 128.112.139.71 -> 63.118.7.16  (atom 12322)
  1        *              0.542ms
  2    12.123.219.133   6.708ms     (AT&T New York)
  3    12.123.0.101     32.232ms    (AT&T Boston)
  4    63.118.7.1       36.345ms    (atom 12322)
```

The topology information maps the second and third hops to AT&T New York and Boston. From this, we can extract a latency sample for the path $NewYork{\rightarrow}Boston$. While the traceroute does not reach the exact destination IP, it reaches the same atom. So we can extract latency samples for two destination paths, $NewYork{\rightarrow}Atom12322$ and $Boston{\rightarrow}Atom12322$. In this step, we remove latency samples that are impacted by path asymmetry or overload (Section 4.3).

In the second step, the remaining latency samples of a path, which may come from different sources, are aggregated into one estimate for the path in that cycle. There are many methods to accomplish this, e.g., arithmetic mean, geometric mean, median, etc. We want a method that is robust to both small and large outliers. Following Roughan and Spatscheck [33], we use the median to aggregate samples from the same source and the arithmetic mean to aggregate across sources.

The final step produces an ISP-wide performance measure across a set of paths and cycles. We use two measures, *stretch* and *spread*, which we believe to be of broad interest; users can also access the output of the second step (see below) to perform this aggregation as they desire. We first aggregate across cycles and produce two values per path. The *stretch* of a path is the additional latency compared to that of a hypothetical direct fiber link between its two end points. For instance, the stretch is 10 ms if the latency is 40 ms for a path between cities whose direct-link distance is 30 ms. As representative latency of a path, we use the trimmed mean, which is the mean of the latency estimates between the 10th and 90th percentile. Using *stretch* instead of absolute latency enables aggregation across paths with different distances between their end-points. The *spread* of a path captures latency variation, using the difference between the 90th and 10th percentile latency samples. We then aggregate across paths using the arithmetic mean of *stretch* and *spread* of individual paths.

5.3 Web Interface

The Netdiff data are publicly available at http://www.netdiff.org/ in two forms. Users who desire a detailed analysis based on their workload can download and process per-path latency estimates. Additionally, we have built an interface to support queries that are likely to be common. Feedback from a real consumer (Section 6.5) suggests that users will often be interested in aggregation based on geography and observing the historical performance of paths of interest. Our interface supports these queries.

Our Web interface makes the data available to users a few hours after the measurements are conducted. We wanted to make a cycle's data available right after it ends, but this task was complicated by the fact that sometimes the results from a prober cannot be retrieved immediately after the cycle. To avoid the need for running the analysis repeatedly as new results trickle in, we wait for all the result files to be retrieved. Maximum waiting time is set to six hours. In the future, we plan to make our analysis engine operate incrementally on new results, which would enable us to provide results in almost real-time.

6 System Evaluation

We have instantiated Netdiff on roughly 700 PlanetLab nodes, which are spread across roughly 300 sites. It has been operational since February 2007. In this section, we use this instantiation to evaluate its design. The next section compares ISPs.

We consider the following questions in subsequent subsections. $i)$ By how much do our optimizations reduce the probing requirement? $ii)$ What is the extent of coverage that Netdiff achieves for ISPs with the current set of nodes? $iii)$ Are Netdiff's path latency estimates reliable? $iv)$ How do its latency estimates relate to application performance? $v)$ Are consumers likely to find Netdiff useful?

Table 1 lists the ISPs that Netdiff currently measures. All of these are major backbone ISPs with different primary geographic regions of operation. While which ISPs are "tier 1" is always open to debate, our list is a superset of the ten tier 1 ISPs as per one source [38]. The second column shows the number of cities in which the ISP has a PoP according to our data. Some ISP cities may be missing. However, past work shows that, even with far fewer than 300 sites, almost all cities are captured using measurements that are similar to ours [23, 37].

6.1 Probing Requirement

The third column of Table 1 shows the average number of paths that Netdiff measures for each ISP. It shows in parenthesis the multiplicative reduction brought by our set covering-packing based optimization. We see that the optimization reduces probes by roughly an order of magnitude. Of the two other probe reduction techniques, using BGP atoms brings roughly a four-fold reduction, from 250K IP prefixes to 55K atoms. Using routing history to select probes brings another order of magnitude reduction, from 16,500K (300 × 55K) probes to fewer than 1,200K on average. Because these reductions multiply, the three methods together reduce probing requirement by a factor of 400.

Observe from the table that the number of paths probed for an ISP is not simply a function of the number of cities.

					Keynote
ISP	**# cities**	**# probes (K) (reduction factor)**	**# destination paths (% of total)**	**# internal paths (% of total)**	**# internal paths (% of Netdiff)**
AOL Transit	27	5 (5.5)	658 (0.05)	230 (32.8)	n/a
AT&T	113	86 (13.5)	13364 (0.24)	838 (6.6)	72 (8.6)
AboveNet	20	40 (7.6)	17258 (1.73)	277 (72.8)	n/a
British Telecom	32	11 (13.7)	4898 (0.31)	440 (44.3)	2 (0.5)
Broadwing	23	28 (9.5)	7655 (0.67)	149 (29.3)	n/a
Cogent	72	161 (10.2)	42620 (1.18)	1799 (35.2)	12 (0.7)
Deutsche Telekom	67	29 (7.7)	2266 (0.07)	129 (2.9)	0 (0.0)
France Telecom	25	31 (12.1)	6092 (0.49)	229 (38.1)	n/a
Global Crossing	60	143 (9.0)	19082 (0.64)	689 (19.4)	n/a
Level3	62	249 (12.7)	70907 (2.29)	1513 (40.0)	30 (2.0)
NTT/Verio	46	98 (8.4)	28943 (1.26)	535 (25.8)	6 (1.1)
Qwest	52	112 (10.9)	20270 (0.78)	696 (26.2)	30 (4.3)
Savvis	41	56 (8.2)	10012 (0.49)	511 (31.1)	20 (3.9)
Sprint	55	136 (13.1)	36366 (1.32)	1208 (40.7)	20 (1.7)
Tiscali	36	30 (8.8)	5483 (0.30)	325 (25.7)	0 (0.0)
VSNL (Teleglobe)	43	30 (14.4)	5500 (0.26)	542 (30.0)	6 (1.1)
Verizon	161	120 (11.5)	20507 (0.26)	2098 (8.1)	306 (14.6)
XO	47	7 (23.5)	1415 (0.06)	522 (24.1)	2 (0.4)

Table 1: *Backbone ISPs that* Netdiff *currently measures. The contents of the columns are explained in the text.*

It depends on several factors, including the location of the probers and the structure of the ISP network. It is higher for MPLS-based networks such as Level3. With MPLS, probes usually observe only the ingress and egress cities of the ISP network because MPLS hides the intermediate cities that are traversed. Without MPLS, they observe other cities along the path as well. Thus, individual probes provide more information about the ISP network, which helps to reduce the number of probes.

6.2 Path Coverage

Next, we study the number of paths Netdiff measures for an ISP from the current set of sites. We consider both destination paths, which begin at the ISP and end at destination networks, and paths internal to the ISP.

The fourth column in Table 1 shows the coverage for destination paths. The percentage is computed based on the total number of possible destination paths and excludes paths that are filtered due to noise and errors. We can measure thousands of paths for almost all ISPs, which represents 0.05-2.29% of all paths. We find that filtering significantly lowers coverage in our current implementation. We are investigating if probes can be assigned to probers such that they are less likely to be filtered, e.g., based on reverse path lengths.

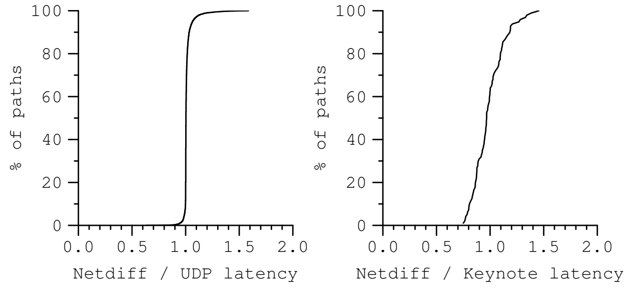
We believe, however, that even the current coverage level of Netdiff is sufficient to derive a reasonable view of an ISP's performance. If our measured paths are not heavily biased, we measure enough paths to obtain representative measures of performance. A similar assumption is used by opinion polls that estimate voting results by sampling a minuscule percentage of the population. While the absence of bias in our measurements is hard to determine with certainty, we conduct several sanity tests to evaluate it. Section 7.3 presents one such test.

The next column shows the coverage for internal paths. It is significantly higher than that of destination paths and varies between 2.9-72.8%. Like the probing requirement, coverage depends on several factors including network topology. For instance, it is low for AT&T because AT&T has a hub-and-spoke topology with many small cities that connect to big cities [37]. Covering paths that begin at smaller cities is harder due to the lack of probers there.

For comparison, we estimate the current coverage of Keynote. Because Keynote does not measure destination paths, we consider only internal paths. It claims to have 1,200 measurement sites worldwide. Based on the list at http://www.keynote.com/support/technical_information/agent_location.html, the last column of Table 1 shows the number of internal paths that it can measure by sending probes between its nodes. The value is 0 for ISPs in which it has only one agent and "n/a" for ISPs in which it has none. Even where Keynote has multiple nodes, its coverage is one to two orders of magnitude less than Netdiff. This difference stems from both our single-ended measurement methodology and that we do not require active cooperation from ISPs. Keynote could in principle start using single-ended measurements, but it would then end up adopting our architecture and would need to address challenges that we tackle in this paper.

6.3 Accuracy of Path Latency Estimates

We now investigate the accuracy of Netdiff's latency measurements to sources of error that include: i) time to gen-



(a) Two-ended control (b) Keynote

Figure 3: *Comparison of* Netdiff *inferences with other methods. Graphs plot CDFs for common paths.*

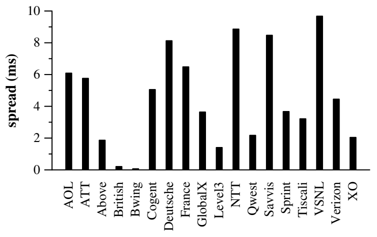


Figure 4: *Average spread in latency measurements of the local path segment.*

erate or forward ICMP responses; ii) path asymmetry; and iii) overloaded PlanetLab nodes or access links. Ideally, we would compare our inferences against an authoritative source. But since we do not have access to such a source, we use four evaluations that together suggest that our latency estimates are reasonably accurate.

Comparison with two-ended control For paths where we can control both ends, we compare Netdiff inferences to latency measured using the more accurate two-ended measurements. The paths we consider in this experiment are those between pairs of PlanetLab nodes. While most such paths traverse the research and education networks, many do not because of PlanetLab sites that are located outside of universities; our results are similar for both kinds of paths. We compare path latencies as measured using Netdiff and using contemporaneous one-way UDP probes in both directions. Because it relies on ICMP responses in the reverse direction, Netdiff will perceive a higher latency than UDP probes if ICMP packets are commonly delayed in transit.

Figure 3(a) shows the relative difference in the latency measurements of UDP probes and Netdiff. We see that the two methods almost always agree. High relative difference in the tail corresponds to very short paths.

Latency variation on the local path segment We now measure the extent of variation in measured latency along the path segment before entering the ISP. This variation can stem from overloaded PlanetLab nodes or overloaded links along this segment. It helps us estimate the impact of local characteristics on Netdiff's measurements.

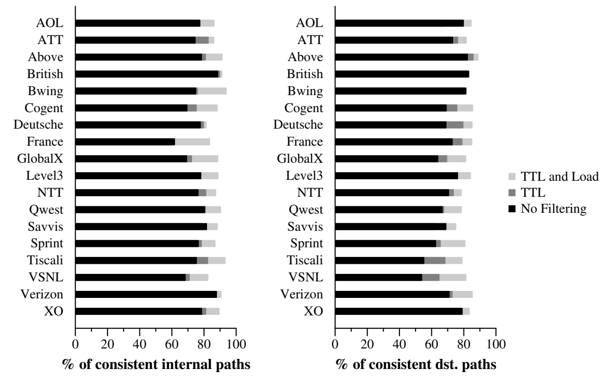


Figure 5: *Consistency of inferences across sources.*

This estimate, however, represents an upper bound on the impact, because Netdiff subtracts the local latency while predicting the latency of the path of interest.

Figure 4 shows that the average latency spread (Section 5.2) for local path segments is typically low, with a maximum value of around 10 ms.

Consistency across sources We now test the consistency of inferences for paths for which we obtain more than one measurement in the same cycle from different nodes. If our estimates are not confused by reverse path asymmetry or local load, these inferences should roughly agree. We consider multiple inferences of a path to be consistent if *all* of them lie within 10 ms. Given that average latency of paths in our data is significantly higher and local variation is within 10 ms, this is a reasonably strong test of consistency.

Figure 5 shows the percentage of paths with consistent measurements. It also shows the percentage for the cases where no filtering is done and where only hop-length-based filtering is done. With both filtering methods, 80-90% of the paths are consistent. We do not expect a complete agreement because the path may be measured at different times. The high consistency level suggests that our inferences are not heavily impacted by noise.

Comparison with Keynote Finally, we compare our latency estimates with Keynote [20] for the paths that are common to both systems. We expect Keynote measurements to be accurate because of its simpler methodology. We compare measurements conducted by the two systems on the same day, but they may not be exactly contemporaneous. If both systems reflect the latency of the underlying path, there should be a rough agreement between the two. On the other hand, if Netdiff estimates are impacted by any source of noise, e.g., ICMP response generation time, they would differ. Figure 3(b) plots the CDF of the average latency estimate of Netdiff divided by that of Keynote. We see that the two systems roughly agree. For 75% of the paths, the relative difference is within 20%. We also study absolute difference and find

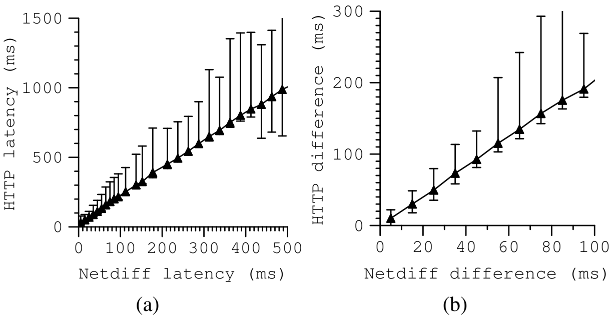


Figure 6: (a) *Relationship between the latency measured by* Netdiff *and the completion time for HTTP transaction.* (b) *The additional time it takes to complete an HTTP transaction as a function of the difference in path latency. In both graphs, the lines connect the medians and the whiskers show the inter-quartile range.*

that 90% of the paths are within 10 ms (not shown).

6.4 Correlation with Application Performance

We study the relationship between path latency measured by Netdiff and application performance. As the application, we consider HTTP transactions to Web servers and quantify performance as the time to complete the transaction. We start with 336K unique web servers visited by CoDeeN [11] users over a two-week period. We map each server to a BGP atom using its IP address. If multiple servers map to the same atom, we pick one randomly. This process yields a list of 8K web servers, each in a different atom. We measure path latency from all sources to these atoms using Netdiff. Contemporaneously, we download the default pages of these servers and log the time to complete each transaction.

Figure 6(a) shows the relationship between path latency measured by Netdiff and the time to complete an HTTP transaction. While HTTP transaction time is a complex function of not only path latency but also loss rate, server load, and page size, our results show that it is strongly correlated with latency estimates of Netdiff.

For pairs of paths to the same server, Figure 6(b) shows the additional time to complete an HTTP transaction along the longer path as a function of the latency difference measured using Netdiff. This further confirms that application performance would be poorer along paths that Netdiff predicts to have higher latency. Figure 6(b) also serves as a guideline for consumers of Netdiff analysis. For instance, if paths through two ISPs differ by 30 ms, the HTTP transaction times will typically differ by roughly 60 ms for small transfers that we use in this experiment and likely more for bigger transfers.

6.5 A Case Study on Usefulness to Customers

We gave early access to Netdiff inferences to operators of a large content provider and asked for their opinion. This provider operates several data centers across the world and connects to many large ISPs. We summarize the operators' views here. This is not meant to be a scientific evaluation but highlights the strengths and limitations of Netdiff from the perspective of a real consumer.

The operators found the capabilities of Netdiff to be useful and novel (despite the fact that they are already customers of Keynote). They especially valued that they could determine the performance of an ISP from a data center city to various destinations. Netdiff lets them do this without signing new contracts with ISPs that they do not connect to and without changing their routing decisions. Of the many ways of observing Netdiff data, the most useful ones to them were being able to see performance based on geography and variations across time.

The operators also pointed out two capabilities that Netdiff does not currently possess but they would find useful. They wanted a close to current view of ISP performance, to aid performance troubleshooting. And they wanted to compare regional ISPs to big backbones for traffic that stays within a region. While these usage scenarios were not part of our original goal, they point at useful directions in which Netdiff can be extended.

7 ISP Comparison Results

We now present a series of results that compare ISPs in different ways. We begin with a comparison that can be considered as indicative of the overall quality of an ISP. Because this hides many differences that are of interest to individual customers and applications, we then compare ISPs in more detail by focusing on specific workloads. Our study is not exhaustive but highlights the kinds of detailed insights on ISP performance that Netdiff can provide. To our knowledge, such detailed information on ISPs' performance was not previously available.

The results below are for a month-long period between Feb. 13 - Mar. 14, 2007. In this period, Netdiff ran continuously without any unplanned or planned disturbance (e.g., for validation). Because of space constraints, we choose to focus on ISPs' performance averaged over the entire period rather than shorter-term variations.

Our results are of course limited to paths that we can measure using PlanetLab. Our sanity tests, however, show that the results are robust to the exact choice of paths. For instance, the results do not qualitatively change even if we discard half of the paths in our data. Section 7.3 describes another such test.

Figure 7 shows an example of the format we use to present results. The x-axis represents a performance measure. The y-axis shows the ISPs, sorted from best to worst. The whiskers represent 90% confidence interval around the average. For visual clarity, the x-axis range varies across graphs. To ease visual comparison, we divide ISPs into five roughly equal groups.

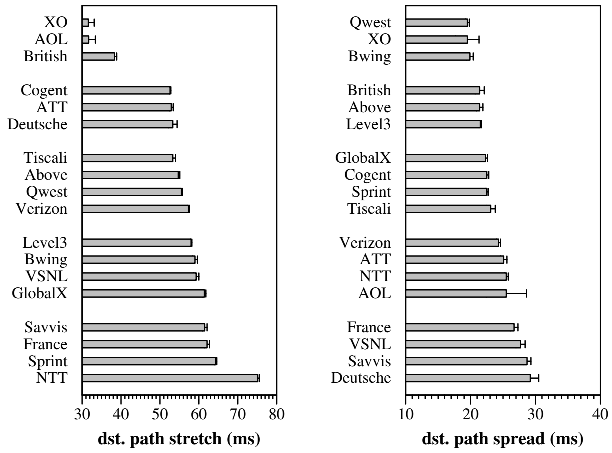


Figure 7: *Stretch and spread of all destination paths.*

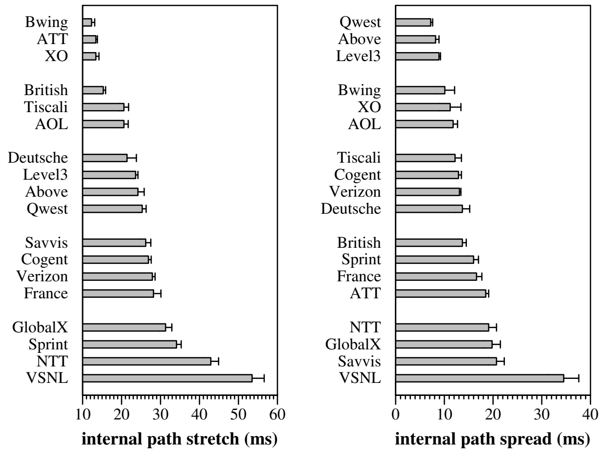


Figure 8: *Stretch and spread of all internal paths.*

Our results quantify the performance of each ISP, and whether the difference between two ISPs is significant is a question that customers must answer based on the needs of their applications. However, based on results in Section 6.3, we recommend that customers ignore performance differences of less than 10 ms between two ISPs.

7.1 Overall Comparison

Figure 7 shows the stretch and spread for all destination paths in our data. It is clear that the choice of ISP is important as the stretch offered by ISPs varies over a wide range. Further, the two measures order the ISPs differently. For instance, Qwest and Bwing have low spread but relatively high stretch. Thus, ISPs that offer the least stretch on average are not necessarily the same as those that also offer consistent path latency.

Figure 8 shows the stretch and spread for all internal paths. These measures provide a different ranking for ISPs. For instance, Bwing has relatively low internal path stretch but a high destination path stretch. Thus, good relative performance for internal paths does not necessarily translate to good relative performance for destination paths. An implication is that ISPs that offer better performance in their SLAs, which typically cover internal paths, may not offer better end-to-end performance.

Such analysis is helpful to not only consumers but also ISPs. For instance, an ISP can tell if problems behind poor performance of destination paths stem from inside its network or outside. For Bwing, for instance, the problems appears to be outside – it has good internal path performance – and changing interdomain neighbors and routing may improve performance.

Results aggregated across all paths hide many differences among ISPs that are relevant to consumers. The rest of this section compares ISPs in more detail.

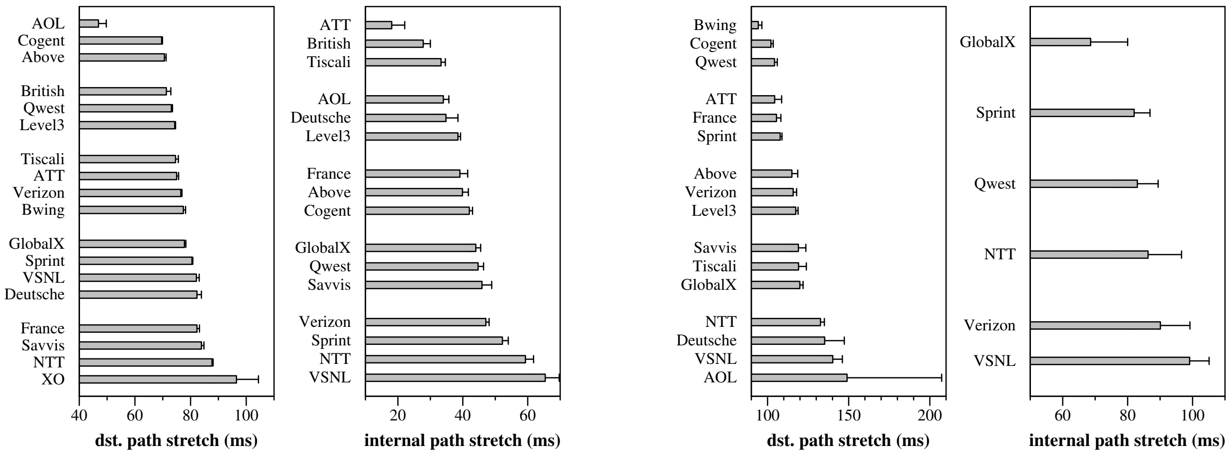
7.2 Dependence on Distance

The first dimension that we study is the distance between the end points of paths. We divide paths into three groups based on the direct hypothetical link distance: $i)$ *short:* less than 20 ms; $ii)$ *medium:* 20-50 ms; and $iii)$ *long:* more than 50 ms. Roughly, long paths are inter-continental, medium-length paths span a continent, and short paths are regional. Figure 9 shows the stretch for medium-length and long paths; for space constraints we omit short paths, which produce a different ordering for ISPs. The missing ISPs in a graph are those for which we have less than ten paths in that category. Many ISPs are missing in Figure 9(b), for long, internal paths, because few ISPs have inter-continental networks.

We see that stretch increases with path length and the relative performance of ISPs differs. While some ISPs are consistently good or bad per our measures, the relative quality of others varies. For instance, Bwing is in the top group for long destination paths but in the third group for medium-length paths. Performance for internal paths suggests that some ISPs are better at carrying traffic internally over shorter distances, while others are better over longer distances.

The six ISPs in the graph for long, internal paths have an inter-continental network. Interestingly, there appears to be little correlation with having an inter-continental network and the performance seen by their consumers for long, destination paths. Based on work that highlights that end-to-end paths can be long due to inter-ISP routing [34], we expected such ISPs to be better at delivering traffic to distant destinations because of potentially fewer inter-ISP transfers. But our results do not bear this out.

The generally higher internal path stretch for these six ISPs in Figure 8 – they all are in the bottom half – might tempt some to conclude that these ISPs are poor. But this is another instance of how judging an ISP only by its



(a) Medium-length paths

(b) Long paths

Figure 9: *Stretch of medium-length (20-50ms) and long (over 50ms) destination and internal paths.*

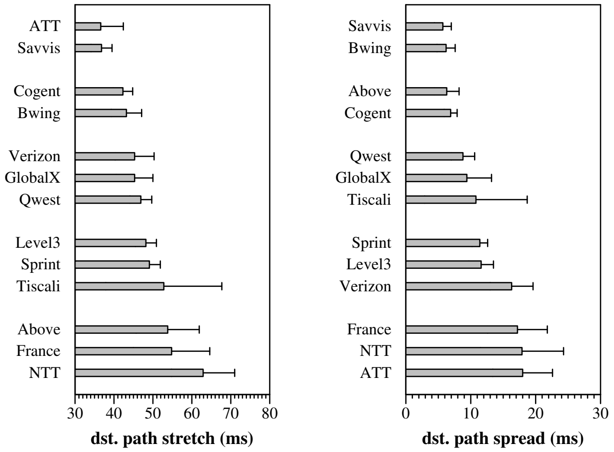
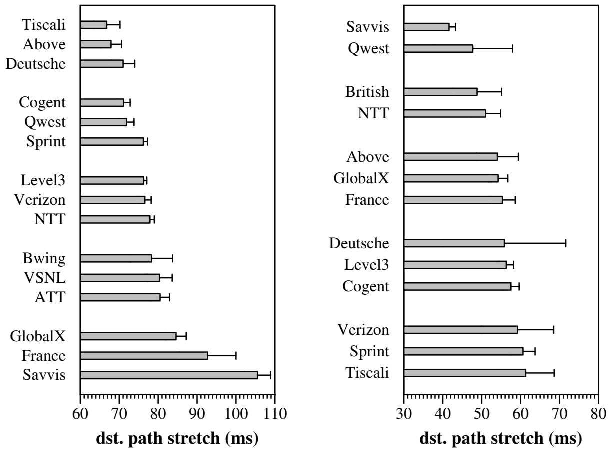


Figure 10: *Stretch and spread for medium-length paths that originate and terminate in the USA.*



(a) From Los Angeles

(b) To Los Angeles

Figure 11: *Stretch for medium-length destination paths that begin near Los Angeles and that terminate there.*

internal paths or SLAs can be misleading. Our analysis shows that the higher internal stretch is simply reflective of their network size and not performance.

7.3 Dependence on Geographic Region

The second dimension that we study is dependence on geographic properties of the traffic. Many consumers will be interested in how an ISP delivers traffic to or from specific regions. We use two example scenarios to show that such consumers may make different choices than location-agnostic consumers.

First, consider consumers that are interested in only one country, perhaps because all of their important nodes reside there. Figure 10 plots the stretch and spread for medium-length paths that originate and terminate inside the USA. Based on the observations in the last section, we do not combine all path lengths. The relative ranking for this case is different than for all medium-length paths in Figure 9(a). For instance, Savvis, which was in the bottom group, is now in the top group. AboveNet moves in the opposite direction. Thus, consumer should make ISP choices based on whether their destinations are mostly domestic or international.

Second, consider consumers that are interested in paths originating or terminating in a specific geographic region. For this, we fold the cities in our data into metropolitan areas because different ISPs may use different city names for the same geographical region (e.g., San Jose versus Mountain View in California, USA). Starting with the list of all cities, we repeatedly select the city with most IPs and include in its metropolitan area all other cities that are within a 100-mile radius.

Figure 11 shows the results for the Los Angeles metropolitan area which is one of the biggest in our analysis. The graphs plot the stretch for medium-length paths originating or terminating near Los Angeles. Some of the

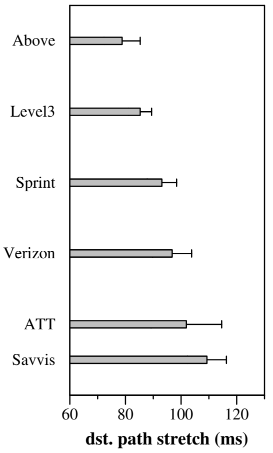


Figure 12: *Stretch for medium-length destination paths from Los Angeles to destinations that are common to ISPs.*

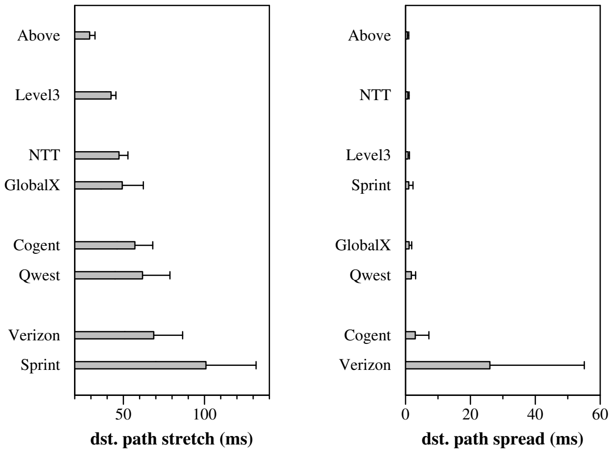


Figure 13: *Stretch and spread for medium-length destination paths to popular web sites.*

ISPs in this figure (e.g., Deutsche Telekom) may not offer a transit service along these paths. For them, the graphs capture performance that customers would experience if the ISP were to offer the service.

We see again that the order of ISPs is different from that for the case of all medium-length paths in Figure 9(a). For instance, Tiscali is significantly better for traffic originating near Los Angeles. Additionally, the relative performance of ISPs for traffic from Los Angeles is different from traffic to Los Angeles, likely because of early exit routing practice by which ISPs transfer traffic to the next ISPs at the closest inter-connection.

A sanity test To test if our results are sensitive to the set of paths that we study, we analyze the ranking for paths from Los Angeles but only to destinations that are common across ISPs. This tests, for instance, whether the differences among ISPs that we find is only because we measure different destinations through them. To ensure we have enough destinations for analysis, we consider only a subset of the ISPs with many common destinations. Figure 12 shows that the relative order of these ISPs is fairly consistent with Figure 11 even though the number of paths being considered is reduced to 6% for some of the ISPs. The only change is the inversion of the order of Sprint and Level3, which have similar stretch.

7.4 Dependence on Destinations

We now study dependence on properties of destinations. So far, we have considered paths to arbitrary destinations, which is more likely to be of interest to content providers. Other consumers may be interested in a specific types of destinations. For instance, a broadband provider may be interested in performance to popular websites. For this experiment, we consider the list of top 100 websites [4] as destinations. Figure 13 shows the ordering for ISPs for which we have enough paths to such destinations. Compared to Figure 9(a), the order changes significantly for some of the ISPs. For instance, for stretch, NTT moves up and Cogent and Qwest move down. We can also see that the performance to popular destinations is in general better than that to arbitrary destinations.

7.5 Summary

Our results show that ISPs differ in various ways, and they underscore the value of Netdiff because it enables customers and applications to pick the ISP that is best suited to deliver their traffic. As a concrete example, it can help a content provider in Los Angeles determine which ISP to use to send traffic to users in East Asia. For this, the provider can use our Web interface to determine the recent and historical performance of various ISPs in carrying traffic from Los Angeles to major cities in East Asia. If it is interested in specific destination networks, e.g., a major broadband service provider, it can use our inferences to make that determination as well. Once it has decided, perhaps after also accounting for cost, it can buy service from the chosen ISP and use it for traffic to East Asia.

8 Conclusions and Future Work

We built Netdiff, a system to uncover detailed performance differences among ISPs. Our work shows that it is possible to build such a system in a way that is easy to deploy, does not require active cooperation from ISPs, and has acceptable probing cost.

Our analysis revealed that the choice of the ISP can significantly impact application performance. But the relative ranking of ISPs depends on many factors, including the distance traveled by traffic and its geographic properties. It also revealed that application performance is not directly reflected in the quality of an ISP's internal paths, the basis of typical SLAs today. Thus, the choice of ISP is a complicated decision that should be based on the properties of the workload. It is in this complex task that Netdiff helps consumers by enabling a detailed analysis of ISP performance.

This paper lays the foundation for our broader goal of objective and comprehensive comparison between ISPs. An obvious direction to extend Netdiff is to measure performance aspects beyond path latency. Single-ended, hop-by-hop measurement techniques for path loss, capacity, and bottleneck bandwidth [16, 18, 28] fit well within our current measurement framework. Another direction is to investigate why two ISPs perform differently. A starting point for this task is to correlate an ISP's performance to its internal network structure as well as to its peering and routing policies. Finally, Netdiff can be extended to compare behaviors beyond performance. For instance, we have started investigating if an ISP's "neutrality" can be measured by studying if they favor or disfavor certain types of traffic.

Acknowledgments We thank the NSDI reviewers and our shepherd, Krishna Gummadi, for their feedback. This work was supported in part by NSF Grants ANI-0335214, CNS-0439842, and CNS-0520053.

References

[1] Y. Afek, O. Ben-Shalom, and A. Bremler-Barr. On the structure and application of BGP policy atoms. In *IMW*, Nov. 2002.

[2] M. Afergan and J. Wroclawski. On the benefits and feasibility of incentive based routing infrastructure. In *PINS*, Sept. 2004.

[3] A. Akella, S. Seshan, and A. Shaikh. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *USENIX Annual Technical Conference*, June 2004.

[4] Alexa Web Wearch – top 500. http://www.alexa.com/site/ds/top_sites?ts_mode=lang&lang=en.

[5] D. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *SOSP*, Oct. 2001.

[6] K. Argyraki, P. Maniatis, D. Cheriton, and S. Shenker. Providing packet obituaries. In *HotNets*, Nov. 2004.

[7] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *IMC*, Oct. 2006.

[8] A. Broido and kc claffy. Analysis of RouteViews BGP data: Policy atoms. In *Workshop on Network-Related Data Management*, May 2001.

[9] A. Brown and D. A. Patterson. Towards availability benchmarks: A case study of software RAID systems. In *USENIX Annual Technical Conference*, June 2000.

[10] P. M. Chen and D. A. Patterson. A new approach to I/O performance evaluation—self-scaling I/O benchmarks, predicted I/O performence. *ACM TOCS*, 12(4), 1994.

[11] CoDeeN: A CDN on PlanetLab. http://codeen.cs.princeton.edu/.

[12] Shop, compare, and save on Internet service providers. http://www.comparingisps.com.

[13] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: A global Internet host distance estimation service. *IEEE/ACM ToN*, 9(5), 2001.

[14] G. Goddard and R. Vaughn. RouteScience's PathControl. http://www.networkworld.com/reviews/2002/0415rev.html, Apr. 2002.

[15] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *INFOCOM*, Mar. 2000.

[16] N. Hu, L. E. Li, Z. M. Mao, P. Steenkiste, and J. Wang. Locating Internet bottlenecks: Algorithms, measurements, and implications. In *SIGCOMM*, Aug. 2004.

[17] Cheapest Internet connections. http://www.isp-connections.com/index.html.

[18] V. Jacobson. Pathchar. ftp://ftp.ee.lbl.gov/pathchar.

[19] kc claffy, T. E. Monk, and D. McRobb. Internet tomography. In *Nature*, Jan. 1999.

[20] Keynote. Internet Health Report. http://www.internetpulse.net.

[21] S. G. Kolliopoulos and N. E. Young. Approximation algorithms for covering/packing integer programs. *Journal of Computer and System Sciences*, 71(4), 2005.

[22] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. An experimental study of delayed Internet routing convergence. In *SIGCOMM*, Aug. 2000.

[23] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements, Mar. 2003.

[24] P. Laskowski and J. Chuang. Network monitors and contracting systems: Competition and innovation. In *SIGCOMM*, Sept. 2006.

[25] J. Lewis. Cogent service. NANOG mailing list archive: http://www.cctec.com/maillists/nanog/historical/0209/msg00684.html, Sept. 2002.

[26] D. Lundquist. Graphical displays of latency/packet loss for top Internet backbones. NANOG mailing list archive: http://www.cctec.com/maillists/nanog/current/msg03940.html, Sept. 2004.

[27] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. An information plane for distributed services. In *OSDI*, Nov. 2006.

[28] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level Internet path diagnosis. In *SOSP*, Oct. 2003.

[29] MaxMind – GeoIP City geolocation IP address to city. http://www.maxmind.com/app/city.

[30] M. Natu, A. Sethi, R. Hardy, and R. Gopaul. Design approaches for stealthy probing. Technical Report 2007/340, Dept. of CIS, Univ. of Delaware, Oct. 2007.

[31] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis. An architecture for large-scale Internet measurement. *IEEE Communications*, 36(8), Aug. 1998.

[32] S. Ratnasamy, S. Shenker, and S. McCanne. Towards an evolvable Internet architecture. In *SIGCOMM*, Aug. 2005.

[33] M. Roughan and O. Spatscheck. What does the mean mean? In *Int'l Teletraffic Congress (ITC) 18*, 2003.

[34] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of Internet path selection. In *SIGCOMM*, Aug. 1999.

[35] N. Semret. Cogent: Disruptive pricing or disruptive marketing. http://web.invisiblehand.net/wp/IHN-WP-Cogent.html.

[36] SPEC – Standard Performance Evaluation Corporation. http://www.spec.org.

[37] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM ToN*, 12(1), 2004.

[38] Tier 1 network. http://en.wikipedia.org/wiki/Tier_1_carrier.

[39] TPC – Transaction Processing Performance Council. http://www.tpc.org.

[40] T. K. Tsai, R. K. Iyer, and D. Jewitt. An approach towards benchmarking of fault-tolerant commercial systems. In *Symp. on Fault-Tolerant Computing*, June 1996.

[41] X. Yang, D. Clark, and A. Berger. NIRA: A new routing architecture. *IEEE/ACM ToN*, 15(4), 2007.

[42] M. Zeier. Genuity – any good? NANOG mailing list archive: http://www.cctec.com/maillists/nanog/historical/0204/msg00318.html, Apr. 2002.

[43] M. Zhang, Y. Ruan, V. S. Pai, and J. Rexford. How DNS misnaming distorts Internet topology mapping. In *USENIX Annual Technical Conference*, June 2006.