A Performance Analysis of Indirect Routing

Joshua M. Opos, Sriram Ramabhadran, Andrew Terry, Joseph Pasquale,
Alex C. Snoeren, Amin Vahdat

Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404

{jopos, sriram, aterry, pasquale, snoeren, vahdat}@cs.ucsd.edu

Abstract

Indirect routing involves sending messages between Internet end nodes through a specified intermediate node to effect a different end-to-end route than the default "direct" route. Indirect routing offers the potential for significant improvements in throughput performance by exploiting the Internet's richness in throughput diversity. We present a performance analysis of indirect routing, showing that it can result in a 33-49% increase in average throughput performance while incurring low overhead.

1 INTRODUCTION

As the Internet grows, the number of possible paths for Internet traffic between end hosts continues to increase. Given this, we explore the potential for improving end-to-end throughput performance through the use of "indirect" paths during periods of poor performance on the "direct" or default paths. The key is to be able to take advantage of *throughput diversity*, i.e., that there are different paths possessing varying levels of available throughput over time. The overall goal is to find the path with the greatest available throughput at any particular point in time.

There have already been various studies that suggest that routing through indirect paths holds potential for performance improvements [3, 7]. Many of these studies have been limited to synthetic measurements of indirect path metrics. There have also been studies that have shown the power of indirect routing in terms of resiliency as well as delay performance [1, 2, 4, 5, 12].

In this paper, we focus on the potential for throughput performance improvement as a result of indirect routing. We show that throughput diversity does indeed exist, and that it can be exploited, at least for large data transfers (at least multiple megabytes). We conducted an empirical study of indirect routing using PlanetLab as a test-bed. The results show that indirect routing is worth doing 45% of the time, leading to a 33-49% average improvement (increase) in throughput performance, especially for situations where the average throughput of the end-to-end direct route is in the range of 1-3 Mbps. Furthermore, these improvements come at a reasonably low cost with respect to overhead and penalties.

The paper is organized as follows. We describe our methodology in Section II, results in Section III, how to select intermediate nodes in Section IV, related work in Section V, and conclusions in Section VI.

2 METHODOLOGY

2.1 Measurement Framework

For convenience, we use HTTP and its support for partial transfers and proxies to test the potential for indirect routing. An experiment consists of the following steps. A client downloads content of known size from a Web server by sending it an HTTP request. It may use either the "direct" path, in which case the download happens via a single end-to-end TCP connection between the client and the server. Alternatively, it may use an "indirect" path, in which case there is an intermediate overlay node interposed between the client and the server using a proxy.

A key issue is how the client determines whether to use the direct or indirect path, and, if using the indirect path, from which intermediate node of multiple candidates to choose. Central to this decision is how well the client is able to predict the rate of a long-lived TCP transfer using a particular path. A simple solution is to initially download a small amount of data over both the direct and indirect paths, and to use the measured throughputs as predictors of the throughputs for the entire download; whichever path has the higher initial throughput is selected.

1-4244-0910-1/07/$20.00 © 2007 IEEE

The HTTP range request option provides a convenient mechanism to implement this strategy. The client sends a range request for the first *x* bytes of an *n*-byte file through both the direct path to the server, and an indirect path (a single one for simplicity; later we consider multiple paths) to an intermediate node which then also forwards the request to the server. Upon reception of these requests, the server responds by sending the first *x* bytes back to the client via both paths. If the client receives the requested data completely through the indirect path first, it will then request the remaining *n-x* bytes through the indirect path. Had the requested data been received on the direct path first, the client would request the remaining *n-x* bytes directly.

The value of *n* varies with the content being requested (i.e., the file size), while *x* is a strategically chosen value large enough to allow the connection to last beyond and marginalize the initial effects of TCP slow-start, and thus get a good estimate of expected throughput. We experimentally determined that *x* = 100KB produces good estimates.

2.2 Experimental Deployment

We deployed our measurement framework on the PlanetLab [9] wide-area network test-bed, making use of 21 nodes located in the continental US and 22 international nodes, with no two machines being deployed at the same geographic site. We selected popular Web sites – eBay, Google, Microsoft (MSN), and Yahoo – chosen as representative of Web sites that serve large amounts of traffic, to act as destination server nodes.

We categorize nodes as Low (0-1.5 Mbps), Medium (1.5-3.0 Mbps), or High (> 3.0 Mbps) throughput, based on measured average throughput to the targeted destination Web servers on the direct path. (Note that the IP addresses of each Web server were hard coded into each request so as to ensure the client was communicating with the same Web server each time a request was made.)

We used international nodes as client nodes because they generally fall into the category of Low throughput client nodes. As one might expect, High throughput client nodes do not benefit from indirect routing as much as Low ones, and may even incur severe penalties. We elaborate on this below, where we show quantitatively that Low throughput client nodes stand to gain the most improvement via indirect routing.

We used USA nodes as intermediate nodes because of their superior connectivity to the destination Web servers, all of which are located in the USA, thus increasing the chances that the overlay link between the client and intermediate node will be the bottleneck on the indirect path.

After deploying and activating the forwarding service on each intermediate node, two client processes are started on the client nodes: one that issues requests as would normally be done on the direct path, and the other which does the preliminary download of the first portion and determines whether to use the direct or indirect path. (Here, we present the results of using a single indirect path that we determined *a priori* to be a good one, though not necessarily the best since it is selected statically; later, we describe results based on dynamically selecting the best indirect path from multiple ones).

Both client processes execute concurrently, performing the same actions: downloading a large file from a particular Web site every 6 minutes for 10 hours (i.e., 100 times). The measurements were taken over a two-month period: April and May of 2005. We determine improvement by comparing the throughput of the “selected” path with that of the direct path. The selected path will be either the indirect or direct path, based on the results of the predictor. Thus, a positive improvement can only result from the indirect path having been chosen. However, choosing the indirect path does not necessarily result in a positive improvement. If the prediction is bad, a negative “improvement” can result.

The Web sites chosen for these experiments are eBay.com, Google.com, Microsoft.com, and Yahoo.com. Requests are not made for files smaller than 2 MB to ensure long-lived TCP transfers.

Indirect routing produces a throughput improvement (i.e., the ratio of the difference between the selected path and the direct path throughputs, to direct path throughput) ranging from 33% to 49% on average, depending on the Web site. For more detailed analysis in the remainder of the paper, we focus on the eBay data set, as it contains a much larger number of data points that correspond to transfers through the indirect path than the other data sets, but is representative of all data sets with respect to the behavior profiles and characteristics observed.

3 RESULTS

3.1 Throughput Improvement Distribution

Throughput improvement is the ratio of the difference between the selected path and the direct path throughputs, to direct path throughput, where the selected path may be either the indirect or direct path. Thus, for example, if utilizing the indirect path doubles the throughput of the direct path of a particular transfer, this will result in an improvement of 100%. If, however, utilizing the indirect path halves the throughput, this is considered a negative improvement of -50%.

Fig. 1 shows throughput improvements across all clients. Each data point contributing to the histogram represents a particular client node performing two transfers of a large multi-megabyte file, one through the selected path and the other through the direct path. The average throughput improvement is 49%; the median is 37%. 84% of the data points lie between 0 and 100.

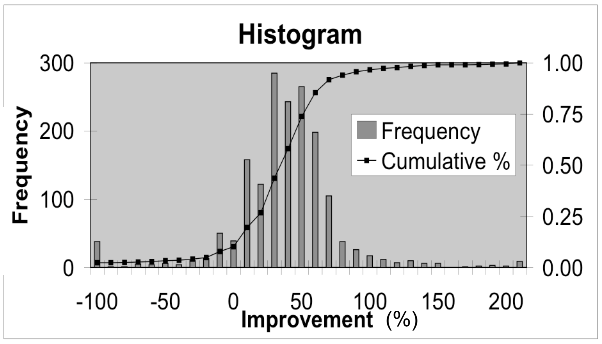

Figure 1. Histogram of throughput improvements aggregated over all clients.

Approximately 12% of the points in Fig. 1 are less than 0, corresponding to a negative performance improvement, or *penalty*. As mentioned above, this can occur if the indirect path is selected because it outperforms the direct path when downloading the initial portion of the file, but not for the transfer of its entirety. In fact, closer inspection of the data shows that these performance penalties correlate to client nodes whose throughputs on the direct path were highly variable. In such cases, it is possible that the indirect path is chosen, but the direct path throughput ultimately becomes much higher for the majority of the transfer.

In fact, by enhancing our collected data with information provided by "traceroute", we observed that these anomalies do in fact occur, but not frequently. That they occur is not surprising; in [11], He et al. show that the throughput of large TCP transfers is highly dependent on path load and the amount of statistical multiplexing on that path, and that these factors can dynamically change throughout the course of a transfer.

Another situation that can lead to performance penalties is when the indirect and direct paths share a common bottleneck link. In this case, the indirect path will suffer from the same problems as the direct path, and will not be able to deliver superior performance. Even if the indirect path slightly outperforms the direct path, the extra overhead of routing through the indirect path will cause the client to incur a performance penalty.

What is promising, however, is that these situations rarely occur on client nodes whose direct paths have more stable throughputs. After conducting a post-hoc analysis of the data in our workload, we found that the majority of the penalties observed involved clients whose throughputs were high on average, thus having a potential for high variability.

Table I shows the results of this analysis. For all clients (clients in all throughput categories: Low, Medium, and High), the fraction of experiments that resulted in penalties was 12%; the average amount of penalty was 290%, with a standard deviation of 706% and a maximum penalty of 3840%.

TABLE I. PENALTY STATISTICS

	Penalty Points	Avg. Penalty	St. Dev.	Max
All	12%	290%	706%	3840%
Med/Low Throughput	8%	43%	71%	356%
Low Variability	3%	12%	7%	35%

We now consider what happens when filtering out points based on client-observed throughputs. First, if we remove those that correlate to High throughput, this reduces the overall number of observed penalties to 8% of the data set, and drastically reduces the average penalty to 43%. Second, if we remove the data points that correlate to Low and Medium throughput client nodes that also have highly variable direct throughputs, this further reduces the overall number of penalties to 3%, and the average penalty to 12%. This, and other data that will be presented later, suggests that the desired situations for indirect routing are Low and Medium throughput client nodes with low throughput variability on the direct path.

The cumulative improvement distribution shown in Fig. 1 captures the profiles of observed per-client distributions collectively. We present some specific per-client histograms in Fig. 2; the separate behaviors of the majority of the client nodes are roughly similar to the cumulative distribution in Fig. 1, in that most of the percent improvement is somewhere between 0% and 100%, and peaks somewhere near 50% (though not in all cases, as with France).

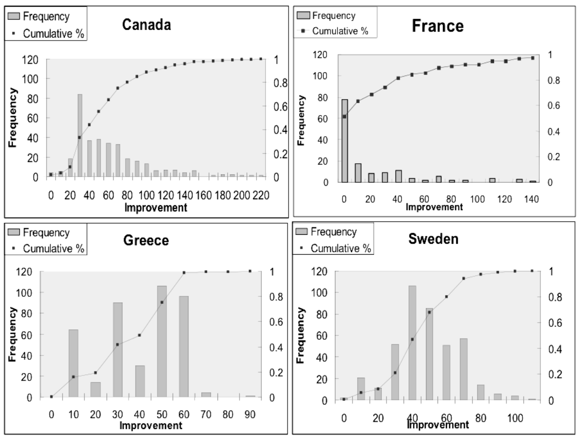

Figure 2. Histograms of throughput improvements for selected clients. Most clients have roughly similar improvement characteristics.

3.2 Intermediate Node Correlation

We now consider which intermediate nodes best serve a particular client node, and how the usefulness of these nodes is distributed across clients. We focus on the question of whether there is a particular intermediate node or set of intermediate nodes that are utilized more often than others when they are on the indirect path.

To answer this question, we turn to Table II, which is a compilation of the three most active intermediate nodes in terms of utilization, i.e., the fraction of total transfers when the indirect path was chosen and while using a particular intermediate node. Table II shows node names and their utilizations in parentheses. For example, for the Australia 2 client, when the indirect path uses Princeton as its intermediate node, the likelihood that the indirect path is chosen is 61%. When Duke is used, the likelihood is 46%, and when Wisconsin is used, the likelihood is 36%. These form the top three intermediate nodes for the Australia 2 client in terms of likelihood of choosing the indirect path.

TABLE II. CLIENT NODES AND THEIR TOP THREE INTERMEDIATE NODES (UTILIZATIONS IN PARENTHESES)

Client	First	Second	Third
Australia 1	Umich (16%)	Wash (8%)	Upenn (8%)
Australia 2	Princeton (61%)	Duke (46%)	Wisc (36%)
Beirut	GTech (88%)	Caltech (68%)	Berkeley (68%)
Berlin	UIUC (62%)	Wash (53%)	NYU (37%)
Brazil	MIT (50%)	Stanford (44%)	UIUC (33%)
Canada	Wash (99%)	Caltech (99%)	Wisc (99%)
Denmark	Upenn (54%)	NYU (35%)	UIUC (18%)
Finland	Upenn (84%)	NYU (84%)	UIUC (26%)
France	Caltech (78%)	Harvard (51%)	ND (41%)
Greece	NYU (99%)	Upenn (97%)	UIUC (93%)
Iceland	Princeton (76%)	ND (73%)	NYU (60%)
India	Berkeley (80%)	Texas (72%)	ND (51%)
Israel	NYU (99%)	Upenn (97%)	UIUC (77%)
Italy	Princeton (99%)	ND (99%)	Harvard (98%)
Korea	Berkeley (50%)	Princeton (38%)	Texas (32%)
Norway	NYU (70%)	Upenn (70%)	Harvard (66%)
Russia	UIUC (83%)	ND (75%)	Wash (41%)
Singapore	CMU (5%)	UCSD (1%)	Caltech (1%)
Sweden	ND (89%)	Princeton (88%)	Harvard (86%)
Switzerland	NYU (65%)	UIUC (63%)	Duke (62%)
Taiwan	UCLA (24%)	Wash (6%)	Wisc (4%)
UK	UCSD (5%)	UCLA (3%)	GTech (1%)

As can be seen, among the top three intermediate nodes for each client, there is a fair amount of overlap, which means that an intermediate node is heavily utilized by more than one client node. This implies that a handful of intermediate nodes may be able to yield a majority of the improvement observed by indirect routing, which holds significant implications for intermediate node selection policies. However, the question remains, why are these nodes so popular in terms of utilization, and what are their common characteristics?

The indirect path is composed of two parts: the path from the client node to the intermediate node, and the path from the intermediate node to the Web server. Since the intermediate nodes are geographically much closer and have much better connectivity to the Web servers (since they are all in the USA), we may assume that, in general, the path between the intermediate nodes and Web servers will have superior throughput, and will not be the bottleneck along the indirect path. Thus, we expect that the bottleneck is the path between the client and intermediate nodes. This implies that the set of intermediate nodes that are heavily utilized across all client nodes have better connectivity on the path from the client to the intermediate node relative to the rest of the intermediate nodes. Therefore, in choosing intermediate nodes, it may be beneficial to select them based on overlay link throughput, if this information is available.

3.3 Improvement vs. Client Throughput

In this section, we determine what types of clients as characterized by their throughputs can benefit most from indirect routing. Each graph in Fig. 3 depicts client improvement vs. direct path throughput on a per intermediate-node basis. These graphs were chosen as being representative of the entire data set.

As can be seen, the trend is that throughput performance improvement decreases as client throughput on the direct path increases. In other words, the lower the client throughput on the direct path, the larger the throughput performance improvement obtained via indirect routing.

One hypothesis as to why client-observed throughput is inversely related to throughput performance improvement is that indirect path throughputs remain fairly constant over time. Thus, improvement decreases as client throughput increases simply because the difference between the two shrinks. Further, for high throughput client nodes, this difference has the potential to become negative and therefore, can result in penalties.

To test this hypothesis, indirect path throughputs were measured over time; each time a client node performed a transfer on the indirect path, throughput was measured. The results are shown in Fig. 4.

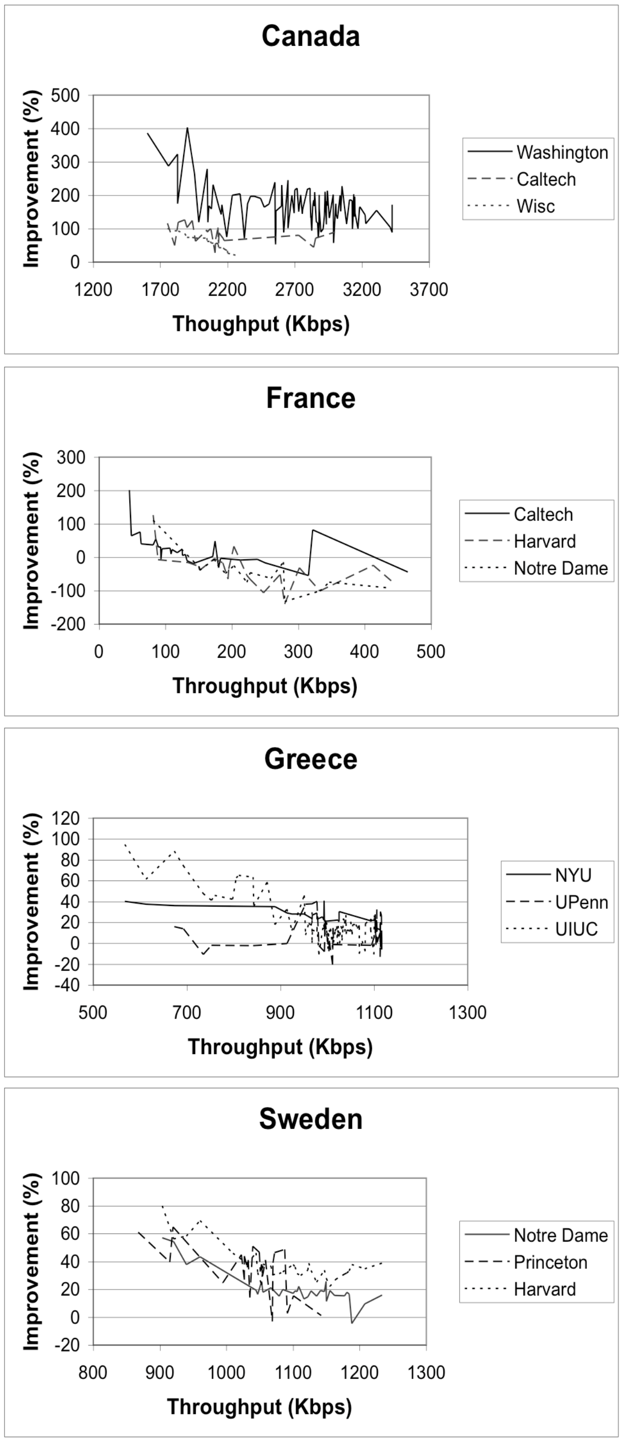

Figure 3. Improvement vs. throughput for selected clients using their top three intermediate nodes. The downward trends suggest that improvement is inversely related to client throughput.

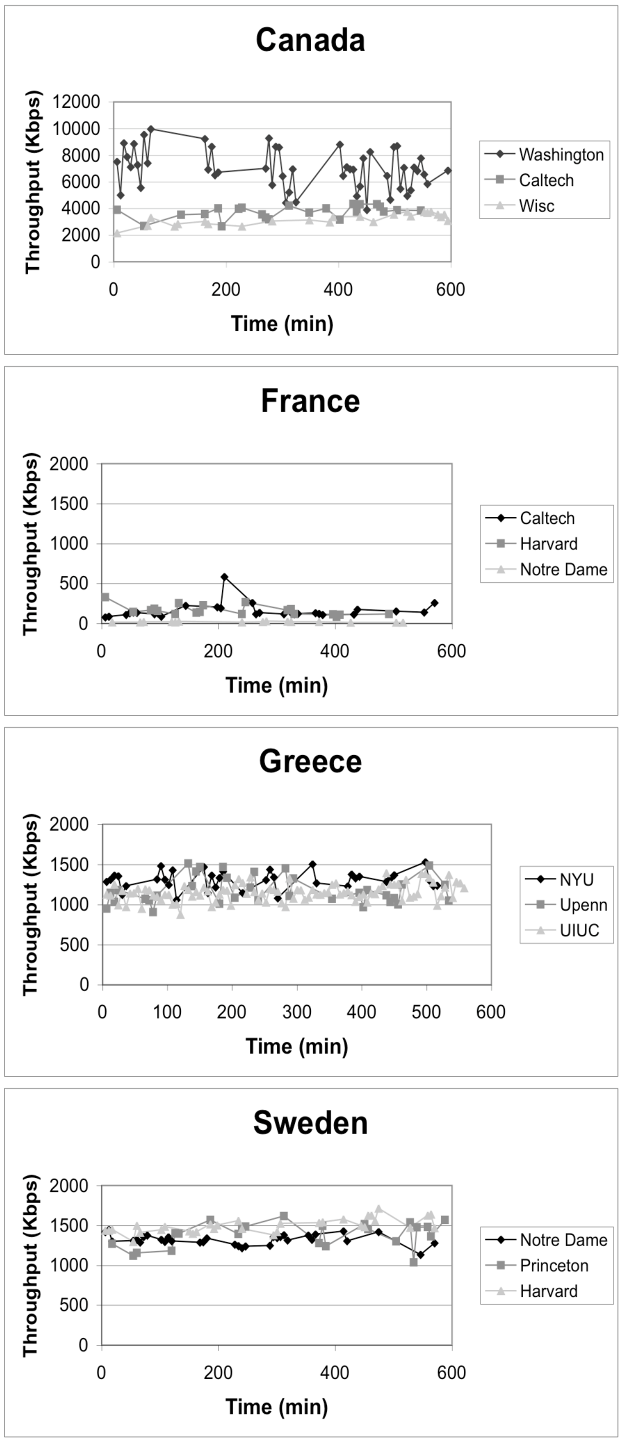

Figure 4. Indirect path throughput vs. time for selected clients. While there are variations over time, there is no discernable uptrend or downtrend.

Indirect path throughputs do not show any discernable uptrend or downtrend over time. However, there are a few small jumps that do occur, which explain why some penalties occur. As discussed above, penalties mostly result from throughput variability. This variability is not limited to the client node throughput. It may be possible for the direct throughput to be somewhat constant, and for the indirect throughput to exhibit superior performance for the transfer of the first range of content, and then sustain a drop once the decision is made to route through the indirect path. These throughput jumps tend to occur far less frequently on the indirect path than on the direct path. Furthermore, note that throughput variability on the indirect path must be the cause of the remainder of the penalties not filtered by the methods described above since all remaining data points correspond to client nodes with invariable direct throughput.

3.4 Indirect Routing Frequencies

We now characterize how often these improvements are achievable, and how often clients, in aggregate, select the indirect path. We define the total utilization of an intermediate node as the number of times an indirect path (using that intermediate node) is actually chosen, divided by the number of times it could have been chosen (over all clients, rather than on a per-client basis as was done for the statistics in Table II).

Fig. 5 shows intermediate node utilizations for a selected set of intermediate nodes chosen to be representative of all of them. To clarify this notion of intermediate node utilization, whenever Berkeley, for example, is the intermediate node of an indirect path, the likelihood that a client will choose the indirect path is 26%. Also shown are the standard deviations and root mean squares, providing a measure of robustness.

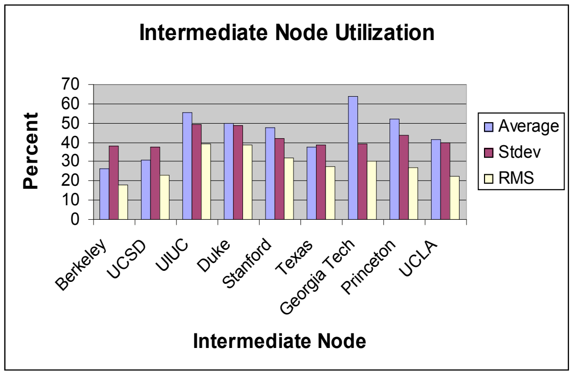

Figure 5. Utilization statistics for selected intermediate nodes.

The average utilization across all intermediate nodes is 45%. This number is substantial and shows that the indirect path is quite active. Implicit in Fig. 5 is the fact that although average utilization varies, the indirect path is still significantly utilized regardless of which intermediate node lies on the indirect path. This implies that there must be significant client throughput variation on the direct path. Different intermediate nodes vary in the levels of available throughput they can provide to a particular client on the indirect path while still being utilized by that client. Therefore, it must be that direct path throughput levels vary, causing transfers to be routed through the indirect path with different frequencies based on available indirect path throughput. The higher the indirect path throughput relative to that of the direct path, the more frequently transfers are routed through the indirect path. Direct throughput variation can cause this relative difference to increase, and sometimes be positive even for an indirect path that may often offer poor throughput.

4 INTERMEDIATE NODE SELECTION

We now consider the question of how a particular client node should select one of possibly many intermediate nodes to effect the indirect path used for a particular transfer. Due to changing network conditions, the best indirect path at different points in time may be different and may depend on the destination. Therefore, it is necessary to have a general, adaptive policy to select an intermediate node from the set of nodes available to a client.

4.1 Selection Framework

In the experiments that follow, a client now has the ability to select from multiple intermediate nodes. The set of all possible intermediate nodes is called the *full set*. For each transfer, a client will use a subset of nodes from the full set from which to choose the actual intermediate node for that transfer. This subset is randomly constructed from the full set using a simple uniform distribution. We will call this subset the *random set*. We seek to determine how large the random set must be to achieve good performance (or whether, say, using the full set is necessary).

4.2 Methodology

We conducted our experiments using a PlanetLab test-bed consisting of 38 nodes; their URLs are contained in Tables IV and V in the Appendix. Three nodes – Duke, Italy, and Sweden – were used as clients, and the other 35 nodes were used as the intermediate nodes (see Table III). These clients were chosen because they are in the Low or Medium throughput categories. In all cases, the destination Web server is eBay.

The setup is similar to that used in the previous experiments with one important difference: two processes run on each client, one operating normally, always making requests on the direct path, while the other is able to select either the direct path or *one of many* indirect paths. The two processes execute closely in time so that time-of-day effects are minimized, but not so closely that they interfere with each other, downloading the same file from the same Web site every 30 seconds for 6 hours (720 times). Each of these 6-hour tests are run on the three clients multiple times while varying the number of intermediate nodes in the random set from 1 to 35. The measurements were taken during May and June of 2005.

Besides measuring the throughput observed by the clients, we also collected data on the usage of each of the intermediate nodes. For each intermediate node, a count of how many times it was selected to be in the random set was recorded. Also maintained was the number of times each intermediate node was actually chosen (over the others in the random set) for transfer over the indirect path.

4.3 Results

First, we determine the optimal size of the random set. We also determine whether an intermediate node that is selected more often than others is also more likely to provide higher performance.

One way for a client to determine the best intermediate node from a random set of size *n* is to perform *n* preliminary download tests (of the first portion of the downloaded file) and see which produces the best throughput. Since this incurs overhead, the smaller the number *n* is, the better; and yet, *n* must be large enough so that the random set offers a selection of intermediate nodes that will result in high throughput. To determine this number, in our tests, we varied the number of intermediate nodes in the random set from 1 to 35.

Fig. 6 shows the average throughput improvements for different random set sizes, where each point represents the average throughput improvement over all 720 transfers. The curves for each of three clients level off at about 10 nodes, suggesting that at least for this group of 35 possible intermediate nodes, a random set size of 10 suffices.

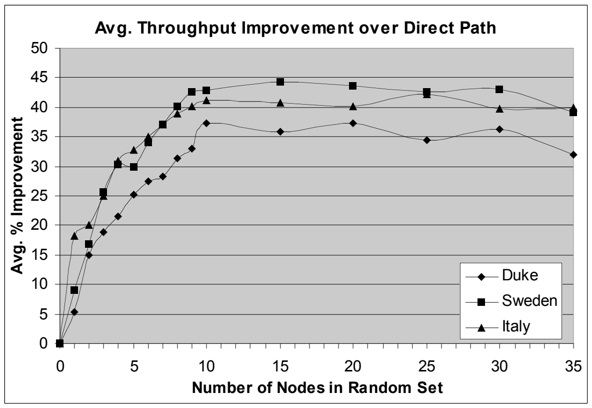

Figure 6. Average improvement in throughput vs. random set size for selected clients (Duke, Sweden, and Italy).

Next we investigate how an intermediate node's utilization corresponds to how well it performs, i.e., how much improvement occurs on the indirect path it affects. As mentioned above, for each intermediate node, we maintain the number of times it was selected to be included in the random set, and the number of times it was selected from the random set as the intermediate node to actually be used for a transfer. In this context, we now define an intermediate node's utilization as the ratio of the number of times it is selected for transfer divided by the number of times that it appears in the random set.

Table III shows the utilizations and the average throughput improvements relative to the direct path for Duke as the client node (which is illustrative of the others, more on this below). Only those intermediate nodes with non-zero utilizations are shown (22 of 35).

TABLE III. INTERMEDIATE NODE UTILIZATIONS AND THROUGHPUT IMPROVEMENTS FOR DUKE AS CLIENT

Node	Utilization (%)	Throughput Improvement (%)
Texas	76.1	71.0
Northwestern	65.9	59.0
Wisconsin	32.9	42.5
Minnesota	32.8	45.7
DePaul	30.2	36.7
Georgia Tech	27.0	28.3
Rice	16.4	13.4
Utah	12.0	31.4
Upenn	8.2	9.2
Maryland	8.1	20.2
Wayne State	5.5	42.1
UCSD	4.2	15.0
Cal Tech	3.8	17.0
UCSB	2.9	11.9
Washington	2.9	3.1
UIUC	2.8	31.1
Berkeley	1.6	3.0
Georgetown	1.5	11.4
Michigan	1.4	27.8
Princeton	1.3	4.9
UCLA	1.3	2.8
MIT	1.3	-19.6

For the most part, the nodes that provide the highest throughput are the nodes that are selected the most. Texas was clearly the best intermediate node, and it is indeed chosen at the highest rate. Notice that this correlation is not perfect. For example, Michigan provides more improvement than many other nodes that are utilized more often. This is because the method of estimating overall throughput by sampling the throughput over the first portion of the file is not a perfect way of making decisions.

Finally, and importantly, while the data in Table III is clearly client specific (e.g., a client different from Duke will have other favored intermediate nodes), it is characteristic of what we observed for other clients. In particular, the general pattern that throughput improvement is correlated with utilization seems to hold across clients.

5 Related Work

Various related works have focused on improved performance through the use of overlay networks. OverQoS [8] is an overlay-based architecture for enhancing the best-effort service of today's Internet. Using a controlled loss virtual link (CLVL) abstraction to bound the loss rate observed by a traffic aggregate, OverQoS can provide a variety of services; the most important relative to our work are statistical throughput guarantees. Guarantees are achieved by statistically bounding the minimum throughput of a bundle (traffic aggregate traveling through a CLVL), offering throughput guarantees to a fraction of OverQoS traffic.

Akella et al. [6] quantify the benefit provided by overlay routing versus BGP with ISP multi-homing route control. The limitations of BGP are often blamed for failures and poor performance of end-to-end transfers, and many studies have shown that these weaknesses can be improved by using overlay routing. Akella et al. show that although overlays do provide better performance, it is not as drastic as previously thought. Overlays are superior when compared with singly homed sites, but sites that use multi-homing only performed slightly worse (5-15% for RTT, 1-10% for throughput). The better performance for overlays comes primarily from their ability to select shorter end-to-end routes because there are a greater number of paths available from which to choose. In our work, we effectively utilize these extra paths to improve throughput performance by means of indirect routing through overlay nodes.

In Bullet [5], Kostic et al. seek to maximize the amount of throughput delivered to multicast receivers, using an overlay mesh that can deliver fundamentally higher throughput and reliability compared to a typical tree structure. Bullet makes data disjoint and distributes it in a uniform way so that the probability of finding a peer containing missing data is equal for all nodes. To find nodes that may have the disjoint data, they developed an algorithm called RanSub [10] that provides a random subset of nodes in an overlay multicast tree once per epoch by performing two passes through the tree (collect and distribute).

The routing flexibility offered by overlay networks can also be used to improve reliability (availability). In a Resilient Overlay Network (RON) [1], nodes regularly monitor the quality and availability of paths to each other, and use this information to dynamically select direct or indirect end-to end paths, leading to significantly improved availability of end-to-end paths between the overlay nodes. The Multi-homed Overlay Network [12] (MONET) system improves client availability to Web sites using a combination of link multi-homing and a cooperative overlay network of peer proxies to obtain a diverse collection of paths between clients and websites. MONET masks failures by obtaining and exploring these different end-to-end paths for each HTTP request made by a client, and was empirically shown to avoid 60-94% of observed failures including access link failures, Internet routing problems, persistent path congestion, and DNS failures, with negligible overhead. One-hop source routing [2] attempts to recover from path failures by routing indirectly through a small set of randomly chosen intermediate nodes, and was empirically shown to support recovery from 56% of network failures.

The Detour project [3, 7] shows that path selection in the wide-area Internet is suboptimal from the standpoint of end-to-end latency, packet loss rate, and TCP throughput. Detour is similar to the work presented in this paper in that it shows the benefits of "detouring" packets via a third node by comparing the long-term average properties of detoured paths to Internet chosen paths. Our paper focuses on the instantaneous properties of such detoured paths, and shows that throughput can be improved on a per-transfer basis using a simple, general framework that can be deployed over any network with cooperating nodes.

Andersen *et al.* [4] note that many routing optimizations that cope with failures, such as overlay networks, are based on an assumption that losses and failures on different network paths are uncorrelated with each other. They show that mesh routing and reactive routing both lower the packet loss rate, and that when both techniques are used together, further improvements are seen, which implies that the two techniques exploit different network properties. They provide many results that show that there is potential for performance-aware indirect routing.

6 Conclusions

There is clear evidence that throughput diversity exists in the Internet, and that indirect routing can take advantage of it to improve throughput. In our study, we have characterized the following properties:

Frequency and magnitude of improvements

The results show that indirect routing can yield an average performance improvement of 33-49%. Furthermore, of the times traffic was routed through the indirect path, positive improvement was observed 88% of the time (while 12% of the time there was a negative improvement).

Clients that stand to gain the most

Clients whose direct path throughputs are in the low-to-medium range stand to gain the most from indirect routing. These clients typically have less throughput variability on the direct path, route through the indirect path more frequently, and experience more improvement while incurring fewer penalties of smaller magnitude relative to high throughput clients.

Client throughput vs. improvement

As an extension of the previous point, we observed that throughput performance improvement is inversely related to client throughput on the direct path. Therefore, as client throughput on the direct path decreases, relative throughput performance improvement generally increases.

Utilization frequency of indirect paths

The average utilization across all intermediate nodes was 45%, showing that selecting the indirect path was significant. Also, the indirect path is used on a consistent basis across client and intermediate nodes, suggesting that the default path may be suboptimal for a significant fraction of time.

Since a positive improvement is observed 88% of the time, and the average amount of time the indirect path is utilized is 45%, we can roughly estimate that throughput diversity can effectively be taken advantage of (to observe positive improvement) approximately 40% of the time for long-lived TCP transfers for the types of clients, servers, and intermediate nodes considered in this study.

Intermediate node selection

We presented an intermediate node selection policy where a client has the ability to monitor the throughput of paths through multiple intermediate nodes. Clients were able to effectively use a random subset of the available intermediate nodes to obtain most of the attainable throughput performance improvement. Since there is a correlation between intermediate node utilization and performance which we did not take advantage of, we expect that higher levels of improvement can be achieved. For example, if a client uses the utilization data to weight the likelihood of a node appearing in the random set, the better nodes will be chosen more often.

IN CONCLUSION, we have shown the potential performance benefits of indirect routing. The effects of poor throughput performance on the direct path can be significantly reduced by routing traffic through an indirect path. Indirect routing can also be used to decrease throughput variability experienced by clients.

REFERENCES

[1] Andersen, D. G., Balakrishnan, H., Kaashoek, F., and Morris, R. "Resilient overlay networks," *Proc. of the 18th ACM Symposium on Operating System Principles (SOSP)*, Banff, Canada, Oct. 2001.

[2] Gummadi, K. P., Madyastha, H.V., Gribble, S. D., Levy, H. M. and Wetherall, D. J. "Improving the reliability of internet paths with one-hop source routing," *Proc. of the 6th Usenix/ACM Operating System Design and Implementation (OSDI)*, San Francisco, CA, December 2004.

[3] Savage, S., Collins, A., Hoffman, E., Snell, J., Anderson, T., "The end-to-end effects of internet path selection," *Proc. of ACM SIGCOMM*, Cambridge, MA, September, 1999.

[4] Andersen, D. G., Snoeren, A.C., Balakrishnan, H. "Best-path vs. multi-path overlay routing," *Proc. of the 3rd ACM SIGCOMM IMC*, Miami, FL, 2003.

[5] Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A., "Bullet: high throughput data dissemination using an overlay mesh," *Proc. of the 19th ACM Symposium on Operating System Principles (SOSP)*, October 2003.

[6] Akella, A., Pang, J., Shaikh, A., Maggs, B., Seshan, S., "A comparison of Ooerlay routing and multihoming route control," *Proc. of ACM SIGCOMM*, Portland, OR. August 2004.

[7] Savage, S., Anderson, T., Aggarwal, A., Becker, D., Cardwell, N., Collins, A., Hoffman, E., Snell, J., Vahdat, A., Voelker, G., Zahorjan, J., "Detour: a case for informed internet routing and transport," *IEEE Micro*, January 1999.

[8] Subramanian, L., Stoica, I., Balakrishnan, H., Katz, R., "OverQoS: an overlay based architecture for enhancing internet QoS," *Proc. of the 1st Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, March 2004.

[9] Peterson, L., Anderson, T., Culler, D., Roscoe, T., "A blueprint for introducing disruptive technology into the internet," *Proc. of ACM HotNets-I*, October 2002.

[10] Kostic, D., Rodriguez, A., Albrecht, J., Bhirud, A., Vahdat, A. "Using random subsets to build scalable network services," *Proc of the 4th USENIX Symposium on Internet Technologies and Systems (USITS),* March 2003

[11] He, Q., Dorvolis, C., Ammar, M., "On the predictability of large transfer TCP throughput," *Proc. of ACM SIGCOMM*, February 2005.

[12] Andersen, D., Balakrishnan, H., Kaashoek, M. F., Rao, R., "Improving web availability for clients with MONET," *Proc of the 2nd Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, May 2005.

APPENDIX

TABLE IV. PLANETLAB CLIENT NODES

	Country	Domain Name
1	Australia 1	plnode02.cs.mu.oz.au
2	Australia 2	planet-lab-1.csse.monash.edu.au
3	Beirut	planetlab1.aub.edu.lb
4	Berlin	planetlab1.info.ucl.ac.be
5	Brazil	planetlab2.lsd.ufcg.edu.br
6	Canada	planetlab1.enel.ucalgary.ca
7	Denmark	planetlab2.diku.dk
8	Finland	planetlab2.hiit.fi
9	France	planetlab2.eurecom.fr
10	Greece	planetlab1.cslab.ece.ntua.gr
11	Iceland	planetlab1.ru.is
12	India	planetlab1.iiitb.ac.in
13	Israel	planetlab2.bgu.ac.il
14	Italy	planetlab1.polito.it
15	Korea	arari.snu.ac.kr
16	Norway	planetlab1.ifi.uio.no
17	Russia	planet-lab.iki.rssi.ru
18	Singapore	soccf-planet-001.comp.nus.edu.sg
19	Sweden	planetlab1.sics.se
20	Switzerland	planetlab02.ethz.ch
21	Taiwan	ent1.cs.nccu.edu.tw
22	UK	planetlab1.rn.informatics.scitech.susx.ac.uk

TABLE V. PLANETLAB INTERMEDIATE NODES

	University	Domain Name
1	CMU	planetlab-2.cmcl.cs.cmu.edu
2	Berkeley	planetlab1.millennium.berkeley.edu
3	Caltech	planlab1.cs.caltech.edu
4	Columbia	planetlab1.comet.columbia.edu
5	Duke	planetlab1.cs.duke.edu
6	Georgia Tech	planet.cc.gt.atl.ga.us
7	Harvard	lefthand.eecs.harvard.edu
8	Michigan	planetlab1.eecs.umich.edu
9	MIT	planetlab1.csail.mit.edu
10	Notre Dame	planetlab1.cse.nd.edu
11	NYU	planet1.scs.cs.nyu.edu
12	Princeton	planetlab-1.cs.princeton.edu
13	Rice	ricepl-1.cs.rice.edu
14	Stanford	planetlab-1.stanford.edu
15	Texas	planetlab1.csres.utexas.edu
16	UCLA	planetlab2.cs.ucla.edu
17	UCSD	planetlab2.ucsd.edu
18	UIUC	planetlab1.cs.uiuc.edu
19	Upenn	planetlab1.cis.upenn.edu
20	Washington	planetlab01.cs.washington.edu
21	Wisconsin	planetlab1.cs.wisc.edu