Some Findings on the Network Performance of Broadband Hosts

Karthik Lakshminarayanan*
University of California at Berkeley
karthik@cs.berkeley.edu

Venkata N. Padmanabhan†
Microsoft Research
padmanab@microsoft.com

ABSTRACT

With the rapid growth in the popularity of and the research interest in peer-to-peer (P2P) systems, an interesting question is what the quality of network connectivity between peers in the "real world" is and what implications this has for applications. In this paper, we describe an effort called *PeerMetric* to directly measure P2P network performance from the vantage point of broadband-connected residential hosts. Our measurements indicate significant asymmetry in bandwidth, with median downstream and upstream available bandwidths of 900 Kbps and 212 Kbps, respectively. We argue that the availability of last-hop bandwidth is more important than the traditional consideration of locality for overlay multicast over broadband hosts. We also consider the peer selection problem and find that a simple delay-vector based approach is effective for finding proximate peers in terms of latency. However, P2P latency turns out to be a poor predictor of P2P TCP throughput, which may be the metric of interest for applications such as file sharing.

Categories Subject Descriptors

C.4 [**Performance of systems**]: Measurement techniques

General Terms

Measurement, Performance

Keywords

Broadband hosts, Network measurement

1. INTRODUCTION

There has been a rapid growth in the popularity of and the research interest in peer-to-peer (P2P) systems and applications. P2P systems have been built for file sharing, content distribution, overlay multicast, etc. While some of the "peers" in these systems may be well-connected machines on academic or enterprise networks[1], a large fraction of them are (or are expected to be) less well-connected machines such as home PCs. An interesting question is what the quality of network connectivity between such "real world" peers is and what implications this has for applications.

While there have been extensive measurement studies of network connectivity and performance between end hosts in the Internet, these have mainly focused on well-connected machines on academic and research networks (e.g., [8, 9]). A few recent efforts have tried to glean information on the network performance of real world peers from measurements of popular P2P applications initiated from well-connected hosts (e.g., [12]). While these efforts have yielded useful information, they have been hampered by their indirect approach; for instance, it has been hard to determine exactly what the latency or TCP throughput between two peers is.

In this paper, we describe *PeerMetric*, an effort we have undertaken to directly measure P2P network performance from the vantage point of broadband-connected residential hosts. This is accomplished by running measurement agents on residential hosts running Microsoft Windows 2000/XP. We considered only broadband hosts (with cable modem or DSL connections) because these constitute a disproportionately large fraction of hosts in P2P systems [12], and this fraction is likely to increase with more widespread deployment of broadband. We deployed PeerMetric on 25 broadband hosts distributed across 9 geographic locations in the U.S. (Figure 1). These hosts were contributed by volunteers and were not (necessarily) members of a real P2P network such as Gnutella. However, given their broadband connectivity, we expect their network performance to be representative of broadband hosts in real P2P systems. So we loosely use the terms "peer" and "P2P" in the context of these hosts.

We gathered a large set of TCP throughput, ping, packet-pair, and traceroute measurements from these vantage points during the period from Sep. 18 through Oct. 13, 2002. There were several questions we sought to answer through these measurements:

Raw performance: What is the bandwidth of broadband hosts and how asymmetric is it? What is P2P latency like?

Peer selection: Is there a quick way to find nearby peers (in terms of network latency) without requiring direct P2P measurements? How good a predictor of P2P TCP throughput are simple ping and packet-pair measurements?

Impact on applications: What implications do these measurements have for applications, in particular overlay multicast?

---

* http://www.cs.berkeley.edu/~karthik/. The author was an intern at Microsoft Research through much of this work.

† http://www.research.microsoft.com/~padmanab/

[1] We use the term "well-connected" to refer to hosts on university or corporate networks that typically have much better connectivity than residential hosts.

IMC'03, October 27–29, 2003, Miami Beach, Florida, USA.

Here are some of our key findings:

- There is a high degree of asymmetry in bandwidth, with the median downstream and upstream available bandwidth (measured as the TCP throughput from and to a well-connected server) being 900 Kbps and 212 Kbps, respectively.
- P2P latencies are much higher than those between well-connected hosts; P2P ping times even within a city are 30-60 ms compared to 3-4 ms between university hosts in similar locations.
- P2P ping time is a poor predictor of P2P TCP throughput, which makes ping time an unattractive metric for peer selection in bandwidth-intensive applications such as file sharing.
- Latency is still important for applications such as P2P search that typically involve exchanging short messages. For these applications, we show that a simple delay-vector based approach [6] is very effective in identifying nearby hosts (in terms of ping time) without requiring direct P2P measurements.
- We argue that the traditional metrics of goodness for application-level multicast (which focus, for instance, on minimizing the use of long-haul, backbone link bandwidth) may be inappropriate in the context of broadband hosts, where the last-hop (upstream) bandwidth is the most constrained resource.

2. RELATED WORK

One of the early sizeable studies of network connectivity and performance between end hosts in the Internet was the network probe daemon (NPD) deployment by Paxson [8, 9]. NPD was deployed at 36 sites worldwide, most of them on academic or research networks. The ability to gather packet-level traces enabled the analysis of phenomena such as packet reordering, which was hard for us to study with PeerMetric. Follow-on efforts such as NIMI [14] have even more extensive deployments but again focus mainly on well-connected sites.

Perhaps the most extensive study to date of real world peers is reported in [12]. This study focused on the hosts participating in the Napster and Gnutella systems. By probing peers from a measurement host at the University of Washington, they measured the latency and bottleneck bandwidth of peers with respect to the measurement host. They report that 50-60% of peers had broadband connectivity; 92% and 78% of peers had a downstream and upstream bottleneck bandwidth, respectively, of at least 100 Kbps; the latency from UW to 20% of peers was under 70 ms and that to another 20% was at least 280 ms. These numbers, however, only offer an indirect indication of the network performance between the peers themselves.

A very recent study [4] has used a similar approach for evaluating various policies for peer selection. Network performance data of roughly 10,000 peers was gathered from 4 measurement points (3 on academic networks and 1 on a DSL connection). While this study reports many interesting findings, it lacks data on the network performance between the peers themselves because all measurements are made with respect to the 4 measurement hosts. So the study is not in a position to answer questions like what the P2P latency or TCP throughput is.

In comparison to previous work, the main distinguishing feature of our work is that we use hosts with broadband connectivity as our measurement points and directly measure the performance between these hosts. On the flip side, however, the logistics of recruiting volunteers to run our software has limited our present study to a modest size of 25 hosts.

3. DESIGN AND IMPLEMENTATION OF PEERMETRIC

The PeerMetric measurement software we built includes a server and a client component. The PeerMetric server is the rendezvous point where clients register their presence when they come online. The PeerMetric client informs the server of its existence by means of periodic keep-alive messages. This soft-state approach makes the system robust to reboots/crashes at the client end. The client supports the following basic tests:

1. Pings/traceroutes to arbitrary Internet hosts
2. Application-level "UDP pings" to other peers (since P2P ICMP pings are often disabled either by NATs or by the peer hosts themselves)
3. UDP packet trains to/from other peers [2]
4. TCP transfers to/from other peers
5. HTTP transfers of specified objects

The logic for deciding which measurement test to invoke, when, and to which target(s) resides in the server. This keeps the client simple and gives us the flexibility to change the schedule of tests as needed.

Due to user privacy concerns, the PeerMetric client does not monitor or record any ongoing activity on the host machine or network. Also, to keep the impact of the PeerMetric measurements on the user's activities minimal, we restricted the volume of measurement traffic at each host to be under 10 Kbps when averaged over a time scale of a few minutes. This limits the rate at which we can initiate tests from or to an individual host (the PeerMetric server honors this limit when issuing tests).

3.1 NAT Traversal

While designing PeerMetric, we had to take special care to traverse NATs since many broadband hosts in the Internet are behind NATs, deployed either at the ISP level or in the home. In our study, 12 of the 25 hosts which were running PeerMetric were behind NATs.

We employed techniques similar to the ones suggested in the IETF STUN [11] proposal for NAT traversal when UDP packets are involved. Since PeerMetric hosts periodically send keep-alive messages to the server, the server is aware of the NAT mapping at these hosts (i.e., the external address that the NAT device maps the internal address of the host to).

There are different types of NATs based on how address mapping and address filtering are performed. *Full cone NATs* map all requests from the same internal address and port to the same external address and port. Any external host can send a packet to a host behind such a NAT by sending to the mapped external address. In contrast, *restricted cone* NATs allow an external host with IP address X and port P to send a packet to the internal host only if the internal host had previously sent a packet to IP address X. In this case, PeerMetric host A behind a NAT is instructed by the server to send a dummy packet to PeerMetric host B. This would set up the necessary mapping at A's NAT to enable B to initiate a measurement back to A[3]. In the case of *symmetric NATs* (which maintain

---

[2] Consecutive packets from a train constitute a "packet-pair" and can be used to estimate the bottleneck bandwidth.

[3] A similar technique can be used for hosts behind *port restricted cone* NATs, which filter incoming packets based on both source address and port

a different external address mapping depending on the destination address and port), we are unable to traverse the NAT in the inbound direction. For hosts behind such NATs, we only report measurements made in the other direction. However, this restriction did not affect our study adversely as only four of the broadband hosts were behind symmetric NATs.

For TCP traffic, the host behind a NAT always opens a TCP connection irrespective of the actual direction of data flow. However, this technique does not work if both hosts are behind NATs. Hence, in our experiments we do not have TCP throughput values for 66 out of the 300 possible host pairs.

4. EXPERIMENTAL RESULTS

We now turn to the motivating questions raised in Section 1: what the raw network performance of broadband hosts (which form the bulk of the peers in real P2P systems) is, what strategies work well for selecting "good" peers in this environment, and what implications our measurements have for P2P applications deployed on broadband hosts (especially compared to accepted wisdom for well-connected hosts). We first describe our measurement methodology and then present our findings.

4.1 Measurement Methodology

We deployed PeerMetric on 25 residential broadband hosts during the period from Sep. 18 through Oct. 13, 2002. These hosts were spread across 9 geographic locations in the U.S. (Figure 1). A total of 8 ISPs are represented in this set. Both the geographic and ISP distributions of the participating hosts were skewed. However, the split between cable modem and DSL connectivity was pretty even with 13 and 12 hosts in the respective categories. We conducted P2P measurements for measuring latency, TCP throughput and bottleneck bandwidth between all pairs of the 25 peer hosts. Note that these P2P measurements correspond to the direct Internet path between the peers, not an overlay path.

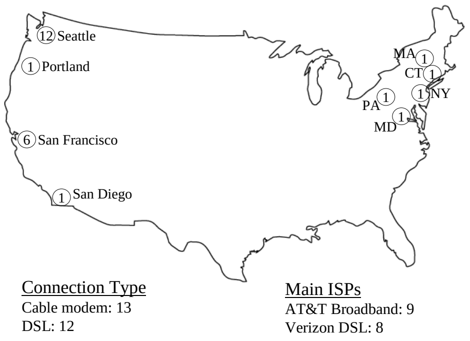

Figure 1: Current deployment of PeerMetric on 25 hosts.

The latency measurements were performed using UDP pings carrying a UDP payload of 16 bytes. Bottleneck bandwidth was estimated by the packet-pair technique [4] using a train of six 1400-byte UDP packets sent back-to-back. Though more sophisticated techniques for estimating bottleneck bandwidth exist (such as [2] and [3]), we believe that this simple technique would work well as we are mostly dealing with relatively low-bandwidth last-hop bottleneck links with little cross traffic. TCP throughput was estimated by performing 100 KB transfers, so chosen to balance accuracy against adverse impact on the network performance of our volunteer residential users. We performed larger TCP transfers to validate the accuracy of the throughput values obtained from 100 KB transfers. It is not surprising that the TCP throughput of 100 KB transfers was close to that obtained by larger transfers given the modest bandwidth-delay product in most cases.

Finally, we compiled a list of 10 well-distributed "landmark" servers in the U.S. and had the peers measure their round-trip time (RTT) with respect to each landmark using ICMP pings.

4.2 Peer Bottleneck Bandwidth

We first study the distribution of the upstream and downstream bottleneck bandwidths for the 25 peers. To estimate the bottleneck bandwidth, we ran packet-pair tests (in both the upstream and downstream directions) between the peers and a well-connected server machine at Microsoft. The underlying assumption is that the bottleneck is at or very close to the last-hop to the peers since the other end is the well-connected server.

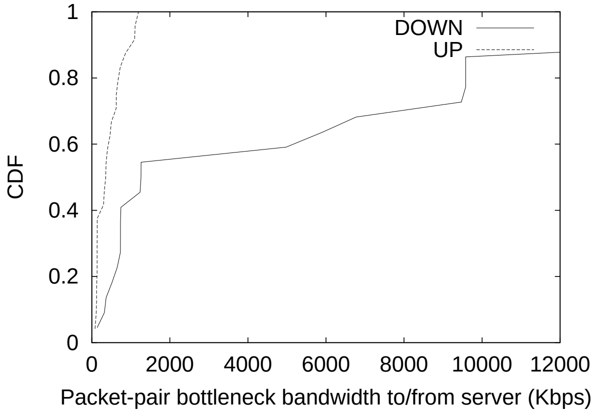

Figure 2: CDF of packet-pair bandwidth estimates for the peers with respect to the well-connected server at Microsoft.

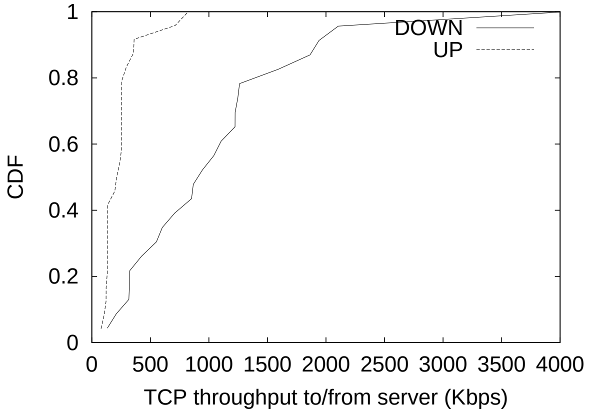

Figure 3: CDF of TCP throughput between the peers and the well-connected server at Microsoft.

Figure 2 shows the CDF of the upstream and downstream packet-pair bottleneck bandwidth estimates for the peers[5]. We notice a significant asymmetry between the upstream and downstream bandwidths. For some peers, the downstream bottleneck bandwidth is

---

[4] Briefly, the idea here is to send a pair of UDP packets back-to-back and measure the spacing between the packets at the receiver. The packet size divided by the spacing observed at the receiver yields a rough estimate of the bottleneck bandwidth (modulo the effects of interfering traffic).

[5] We considered the median measurement for each peer when plotting CDFs, so that the impact of outliers is minimized.

very large (in excess of 10 Mbps). Since this finding is at variance with anecdotal information on the speed of residential broadband connections, we took a closer look at the measurements. We found that all of these apparently anomalous cases corresponded to cable modem hosts (in multiple ISP networks — AT&T Broadband, Comcast, and AOL/TW Roadrunner). Information on how certain commercial cable router products (e.g., the Cisco uBR7200 [13]) do traffic shaping may offer an explanation. Traffic shaping is typically done using a token bucket, which often lets short bursts of packets (e.g., packet-pairs) through without an additional delay introduced between the packets. So the spacing between the packets reflects the raw speed of the wire, not the speed of the link for a sustained data transfer. Clearly, the notion of bottleneck bandwidth needs to be defined carefully in such cases.

To get a more realistic idea of the available bandwidth at the peers, we plot in Figure 3 the CDF of the TCP throughput with respect to the well-connected server. Again we observe significant asymmetry, with median upstream and downstream throughputs of 212 Kbps and 900 Kbps, respectively. This asymmetry is consistent with the findings in [12]. The limited upstream bandwidth could be problematic for P2P applications (e.g., Section 4.5).

4.3 P2P Latency and Throughput

We now turn to measurements of P2P ping times and TCP throughput. The CDFs for these are shown in Figures 4 and 5. We are interested in studying the impact of connectivity type as well as geographic location, so each figure depicts 4 curves — one corresponding to all pairs of peers and one each corresponding to pairs confined to hosts on cable, on DSL, and in Seattle (which had the largest concentration of PeerMetric hosts).

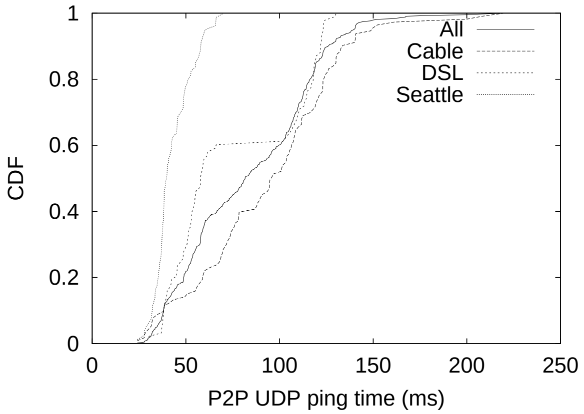

Figure 4: CDF of P2P latency.

From Figure 4 we observe that the latency between hosts in Seattle tends to be significantly smaller than that between arbitrary pairs of hosts (a median P2P ping time of about 40 ms versus 80 ms). Furthermore, the DSL curve shows a marked difference in the latency measured among broadband hosts within the east and west coasts of the U.S. and that between hosts on the opposite coasts. That geographic proximity has a bearing on network proximity is not surprising. Nevertheless, the median latency of 40 ms even among broadband hosts within the same city is an order of magnitude larger than that we measured among well-connected university and corporate hosts in similar locations. Also, the latency among cable hosts (even in the same city) is considerably larger than that between DSL hosts. The shared nature of the cable medium and the contention this entails may explain this larger (and more variable) latency. The considerable latency even among cable hosts in close proximity explains the absence of any noticeable step in the cable curve unlike the one in the DSL curve.

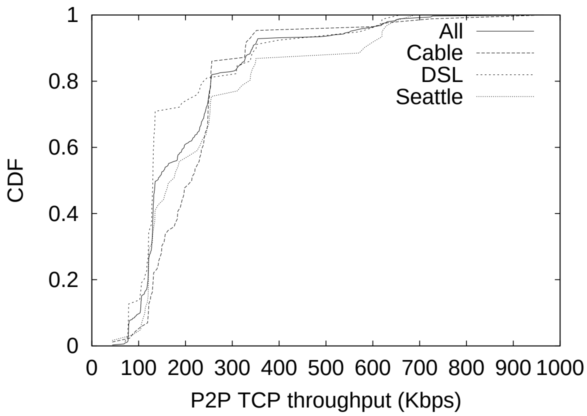

Figure 5: CDF of P2P TCP throughput.

The trends in the case of P2P TCP throughput are quite different (Figure 5). Cable modem hosts outperform DSL hosts (median throughput value of 220 Kbps versus 120 Kbps), and the Seattle hosts exhibit an intermediate level of performance. Thus the trends in P2P latency appear to correlate weakly with the trends in P2P throughput, which we explore further in Section 4.4.2.

4.4 Peer Selection

We now consider the implications of the bandwidth, latency, and throughput measurements presented thus far for the important problem of peer selection. The goal is to enable hosts to find peers to whom they have "good" connectivity. We consider two goodness criteria — low latency and high TCP throughput.

4.4.1 Latency Metric

In certain applications that are not bandwidth intensive (e.g., overlay construction for P2P search), an important question is how to pick peers that are "close" in terms of network latency. While pinging each peer a number of times is a possibility, this is clearly not a scalable approach for all peers to employ. So we consider an alternative where each peer determines its "coordinates" by pinging a fixed set of "landmarks". To find a proximate peer, a host looks for a peer whose coordinates lie near its own coordinates, without requiring any P2P measurements.

The specific approach we investigate is motivated by the *GeoPing* technique previously developed for determining the geographic location of well-connected Internet hosts [6]. Although our interest here is network proximity rather than geographic location, we still use the term GeoPing to refer to the technique. For each peer, we construct a *delay vector* (termed the node's "coordinates") comprising the median delay to each of the 10 well-distributed landmark servers in our list.[6] For each pair of peers, we compute the correlation between the Euclidean distance between their delay vectors and the P2P latency (measured directly by PeerMetric). Since what we are interested in is peer *selection*, we also compute the rank correlation between these two quantities. (The rank correlation only considers the ordering of the peers based on the metric of interest and hence may be more appropriate for the peer selection question.)

Figure 6 depicts a scatter plot of the Euclidean distance between the peers' delay vectors and the measured P2P latency. The good correlation apparent visually is reinforced by high coefficients of linear and rank correlation — both equal to 0.73. So picking peers

---

[6] Having a larger number of landmarks yielded little improvement.

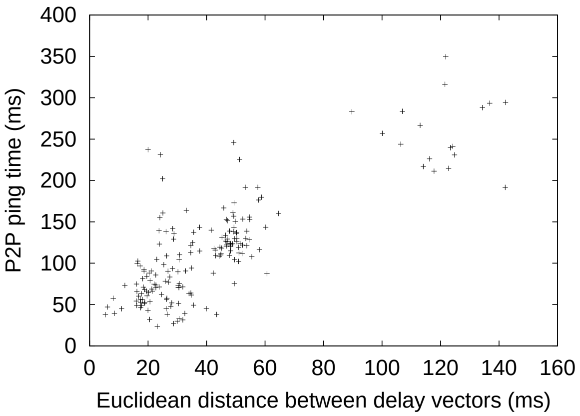

Figure 6: Scatter plot of the Euclidean distance between the peers' delay vectors versus the measured P2P latency.

that are closest in terms of Euclidean distance is a promising way of finding proximate peers.

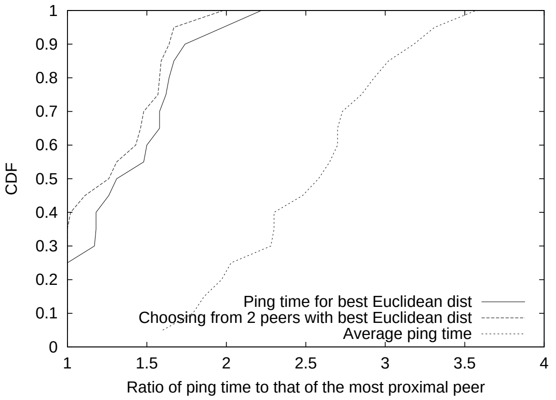

Figure 7: Ratio of the measured ping time to the peer chosen by GeoPing to the ping time to the closest peer.

It is also interesting to evaluate the effectiveness of GeoPing in terms of a metric that applications can directly relate to. We quantify the goodness of the peer picked by GeoPing using the ratio between the measured ping times to the chosen peer and to the closest peer. Figure 7 shows that in 90% of cases, the ping time to the peer picked by GeoPing is within a factor of 1.76 of the ping time to the closest peer. Furthermore, a slightly more heavyweight approach, involving finding the two closest peers reported by GeoPing and picking the better of the two based on measured ping times, results in the chosen peers being within a factor of 1.63 of the optimal choice in 90% of cases. In comparison, the spread in ping time ratios over all peers is considerably larger (as indicated by the "average" curve in Figure 7), which suggests that the lightweight GeoPing technique is quite effective in finding proximate peers. It would be interesting to see how GeoPing compares with the more sophisticated alternatives proposed in [5] and [10].

An application that involves constructing a low-latency P2P overlay network might employ GeoPing as follows. Each potential peer measures and periodically updates its coordinates with respect to a set of landmarks. When it joins the P2P network, it registers its coordinates with a server. The server compares the coordinates of the new peer with those of the previously registered peers and returns a list of one or more (likely) proximal peers.

4.4.2 Throughput Metric

For some applications, such as file sharing, P2P TCP throughput is an important consideration for peer selection. It is desirable for a host to have a quick way of telling which peer is likely to offer the best TCP throughput (say for file download). It has been suggested that picking the "closest" peer in terms of network latency (i.e., ping time) may be a reasonable strategy, in part because of the inverse relationship between the round-trip time (RTT) and TCP throughput. To determine if our data bears this out, we computed the correlation between the median P2P latency and the median P2P throughput. The coefficient of linear correlation was -0.14 and the rank correlation was -0.13. In other words, P2P latency is a poor predictor of P2P throughput. The inverse relationship between RTT and TCP throughput is masked by the wide range in last-hop peer bandwidth, which has little to do with P2P latency. Note that our findings are based solely on broadband hosts; latency may in fact be a good predictor of throughput when dial-up hosts (with large latencies and low bandwidth) are included, which may explain the stronger predictive power of P2P latency reported in [4].

Since obtaining a P2P packet-pair bandwidth estimate is also relatively inexpensive, we investigated how well it correlates with P2P throughput. The coefficient of linear correlation was 0.49 and the rank correlation was 0.75. So despite the problems discussed in Section 4.2, a packet-pair bandwidth estimate is a better predictor of P2P TCP throughput than P2P latency is. We also separately considered pairs of DSL hosts and pairs of cable modem hosts. The coefficient of linear correlation and the rank correlation between the packet-pair bottleneck bandwidth estimate and P2P TCP throughput were 0.79 and 0.92, respectively, in the case of DSL host pairs, and 0.33 and 0.03, respectively, in the case of cable modem pairs. Thus the packet-pair estimate is a good predictor of TCP throughput in the case of DSL hosts but not in the case of cable modem hosts (for the reasons discussed in Section 4.2).

4.5 Multicast tree construction

Finally, using the P2P bandwidth and latency estimates, we try to get an idea of how well an end-system based overlay multicast algorithm (such as [1]) would work when operating over end systems with broadband connectivity. An interesting issue is the trade-off between the achievable bandwidth and the maximum delay that a node may experience. To explore this trade-off, we first fixed a host in Seattle with a symmetric bandwidth of 750 Kbps as the source. For a range of values of the multicast stream bandwidth, we found (using a heuristic search technique) the tree that provided the best "maximum delay" across all nodes (i.e., least delay to the deepest leaf). We only consider traditional single-tree multicast; multi-tree approaches (e.g., CoopNet [7]) could yield better performance.

Figure 8 depicts the trade-off between the stream bandwidth and the best maximum delay. We see that even if the application is willing to tolerate a large maximum delay (over 120 ms), there is not sufficient "outgoing" bandwidth in the system to enable the construction of trees that can support a stream bandwidth larger than 148 kbps. This is primarily because most broadband hosts have a low upstream bandwidth (Section 4.2), which limits the out-degree of the nodes drastically. Furthermore, the maximum delay will only get worse as we scale from 25 peers to 100s or 1000s of peers.

We also studied how well a locality-driven heuristic for tree construction would perform. This heuristic strives to minimize the number of traversals of long-haul Internet backbone links, thereby optimizing the resource usage metrics proposed in application-level multicast research (e.g., [1]). We divided the nodes into 5 clusters based on their locations — East Coast, SF Bay Area, Seattle, San Diego, and Portland. We then considered the subset of trees where

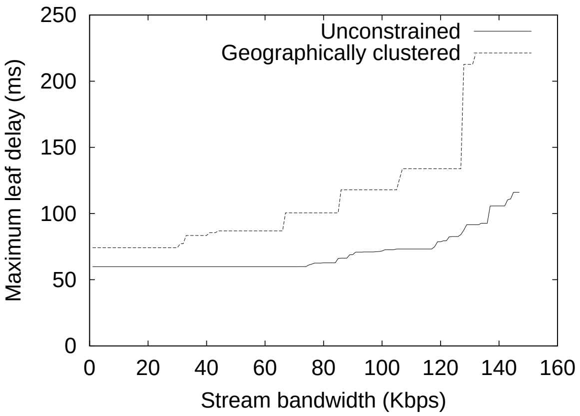

Figure 8: Bandwidth vs. delay trade-off for overlay multicast.

nodes in a cluster are close to each other in the tree (i.e., form a connected sub-graph). We plot the delay corresponding to the best tree obtained given this constraint. We see that the delay for these geographically clustered trees is significantly worse than that for the best possible tree. This is so because in some regions there are senders with high outgoing bandwidth that receivers in other regions do not make use of, thus increasing the depth and the delay of the tree. This is especially so when the group size is not very large, making regional imbalance in the availability of last-hop bandwidth more likely.

This suggests that in the context of broadband hosts it is more important to consider the bandwidth of peers than their location when constructing overlay multicast trees. The conventional wisdom of mimicking native IP multicast by preserving locality in application-level multicast trees may not be appropriate in the context of broadband hosts. The availability of last-hop bandwidth (especially in the upstream direction) is a more important consideration than the usage of long-haul backbone links. So it may well be desirable from a performance viewpoint for multiple hosts in San Francisco to individually connect to parent hosts in New York rather than insist that a single parent-child link traverse the NY-SF backbone.

5. CONCLUSION

In this paper, we have explored the characteristics of network performance among residential broadband hosts (which can be considered representative of peers in the real world) through a modest-sized deployment of our PeerMetric measurement software on 25 geographically distributed hosts. Our motivation is to understand how these characteristics differ from those of well-connected university hosts studied extensively in the literature, and what implications these have for P2P applications.

Our main findings are: (a) The bandwidth of broadband hosts is highly asymmetric (median downstream and upstream available bandwidths of 900 Kbps and 212 Kbps). The limited bandwidth, especially in the upstream direction, makes it the most important consideration for applications such as overlay multicast. (b) For peer selection based on the latency metric (e.g., for constructing a P2P search network), the simple GeoPing technique of constructing and comparing delay vectors is quite effective. (c) For peer selection in cases where TCP throughput is the key metric (e.g., a P2P file sharing application), P2P latency is a poor predictor. The inverse relationship between RTT and throughput predicted by theory is masked by the wide range in last-hop bandwidths. A packet-pair based bottleneck bandwidth estimate, on the other hand, is a good predictor of TCP throughput in the case of DSL hosts. However, packet-pair measurements are unreliable in a cable modem setting, presumably because of the way bandwidth throttling is done.

Acknowledgments

We are indebted to the many friends and colleagues who fearlessly installed PeerMetric on their broadband hosts: A. Adya, V. Alamelu, V. Bahl, W. Chong, P. Chou, R. Draves, K. Gomatam, P. Gopalakrishnan, V. Iyengar, V. Iyer, U. Krishnaswamy, J. Lorch, R. Manian, C. Narayanaswami, J. Padhye, K. Parthasarathy, I. Ramani, A. Rangarajan, S. Ratnasamy, D. Rubenstein, R. Sankaran, M. VanAntwerp, C. Verbowski, and A. Wolman. We would also like to thank G. Nordlund for helping with the network configuration of our server, J. Yagelowich for letting us use his NAT test lab, and A. Adya, J. Padhye and M. VanAntwerp for useful discussions on the design and implementation of PeerMetric.

6. REFERENCES

[1] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang. “Enabling Conferencing Applications on the Internet Using an Overlay Multicast Architecture”, *ACM SIGCOMM*, August 2001.

[2] C. Dovrolis, P. Ramanathan, and D. Moore, “What do Packet Dispersion Techniques Measure?”, *IEEE INFOCOM*, 2001.

[3] K. Lai and M. Baker, “Nettimer: A Tool for Measuring Bottleneck Link Bandwidth”, *USITS*, March 2001.

[4] T. S. E. Ng, Y. Chu, S. G. Rao, K. Sripanidkulchai, and H. Zhang. “Measurement-Based Optimization Techniques for Bandwidth-Demanding Peer-to-Peer Systems”, *IEEE INFOCOM*, March 2003.

[5] T. S. E. Ng and H. Zhang, “Predicting Internet Network Distance with Coordinates-Based Approaches”, *IEEE INFOCOM*, June 2002.

[6] V. N. Padmanabhan and L. Subramanian. “An Investigation of Geographic Mapping Techniques for Internet Hosts”, *ACM SIGCOMM*, August 2001.

[7] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai. “Distributing Streaming Media Content Using Cooperative Networking”, *NOSSDAV*, May, 2002.

[8] V. Paxson. “End-to-End Routing Behavior in the Internet”, *IEEE/ACM Transactions on Networking*, Vol.5, No.5, pp. 601-615, October 1997.

[9] V. Paxson. “End-to-End Internet Packet Dynamics”, *IEEE/ACM Transactions on Networking*, Vol.7, No.3, pp. 277-292, June 1999.

[10] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. “Topologically-Aware Overlay Construction and Server Selection”, *IEEE INFOCOM*, June 2002.

[11] J. Rosenberg, J. Weinberger, C. Huitema and R. Mahy. “STUN - Simple Traversal of UDP Through Network Address Translators”, IETF Draft, December 2002.

[12] S. Saroiu, P. K. Gummadi, and S. D. Gribble. “A Measurement Study of Peer-to-Peer File Sharing Systems”, *MMCN*, January 2002.

[13] Cisco uBR7200 Series Universal Broadband Router, *http://www.cisco.com/warp/public/cc/pd/rt/ub7200/index.shtml*

[14] National Internet Measurement Infrastructure (NIMI), *http://www.ncne.nlanr.net/nimi/*