

pcp: Internet Latency Estimation Using CDN Replicas

Samuel Micka, Utkarsh Goel, Hanlu Ye,[†] Mike P. Wittie, and Brendan Mumey

Department of Computer Science, Montana State University, Bozeman, MT 59717

[†]Department of Computer Science, Wellesley College, Wellesley, MA 02481

{sam.micka, utkarsh.goel, mwittie, mumey}@cs.montana.edu, hye@wellesley.edu

Abstract—Real-time applications have become increasingly common and necessary to perform everyday tasks for many users. These applications speed up their communications by connecting users to the closest servers and clustering nearby users together. In replica server selection and client clustering, determination of the closest host quickly and accurately is crucial to quick delivery of application responses and the satisfaction of users' expectations. Researchers commonly use latency as the primary metric of network proximity and have developed various latency approximation tools. However, these tools do not yet offer an attractive balance of measurement accuracy, scalability, and maintainability. In this paper, we propose a new latency estimation system for arbitrary hosts using host-to-CDN latency measurements. Compared to existing latency estimation tools, our technique offers superior coverage of the IP address space and latency estimation accuracy. With improved coverage and accuracy of latency estimation it will become easier to establish low latency connections between hosts in a network, improving the responsiveness of real-time and interactive Internet applications.

Index Terms—latency estimation, server selection, P2P clustering, network coordinate systems

I. INTRODUCTION

Low latency network communication is prerequisite to a responsive user experience in interactive and real-time Internet applications. In applications such as augmented reality, online gaming, video chat, and cloud-based voice recognition even small communication delays lead to user-perceived loss of responsiveness and, eventually, user attrition and loss of application revenue [1]–[7]. For example, when latency is high in online games player views of the global game state diverge [8]. View inconsistency makes it harder for players to observe and control game actions, which makes it more difficult for them to remain immersed in the game and perform well [9]–[11]. In practice game forums are full of complaints about high latency to game **servers** and to **other players** in games with peer-to-peer hosting models [12]–[14].

Many applications reduce communication latency by directing users to connect with nearby servers and other nearby users. Lower latency connections deliver content more quickly thanks to a faster expansion of the TCP congestion window and more rapid retransmissions over low round-trip time (RTT) paths. Lower latency also speeds up short message exchanges, especially in group applications, where communication rounds may need to wait for the most *lagged* user [15]. Applications of these principles exist in DNS redirection used by CDNs [16] and proxied connections in public cloud networks [17]–[19].

The challenge then is how to accurately and efficiently select a server, or a group of users, closest to a given user. Direct latency probing, for example using `ping`, is accurate, but also time consuming and does not scale [20]. Tools, such as CloudGPS, reduce the number of user measurements, but require cooperation between ISPs and cloud providers [21]. Content distribution networks (CDNs) rely on DNS redirection, though when a user's DNS server is not nearby, for example in the case of public DNS infrastructure, the likelihood of selecting a nearby server is low [22].

Instead the research community has proposed a number of predictive *matchmaking* tools to identify the closest replica server to a client IP based on geographic proximity [23]–[26], or network distance [20], [27]–[30]. However, these tools suffer from incomplete coverage of the IP space, make predictions based on stale network measurements, or are proprietary. Finally, many of these projects are defunct leaving applications with few realistic options for end-point selection.

This paper makes two contributions. First, we undertake a comparative evaluation of **currently available** latency estimation tools [23], [26], [27], [31], [32]. We show that these solutions suffer from low coverage of the IP address space and low latency estimation accuracy, which translates to poor choices of nearby network endpoints. Second, we propose Ping through CDN Proxies (`pcp`) – a new tool for Internet latency estimation. `pcp` approximates the latency between two arbitrary hosts in a network based on the length of a path in a virtual network topology built from measurements between clients and proxy CDN servers. Our extensive evaluation on the Dasu and Seattle network testbeds shows that `pcp` offers more accurate latency approximation than existing tools [33], [34]. `pcp` is also more scalable than `ping` and in fact, unlike existing tools, does not require active probing. Finally, `pcp` is likely to stand the test of time, because it uses CDN servers for its measurement infrastructure and does not require special network coordinate servers to be maintained [27], or ISP cooperation [21].

We expect that `pcp` will provide an accurate, scalable, and long-lived alternative for Internet latency estimation. Improved endpoint selection has the potential to improve interactive and real-time application responsiveness as well as the quality of user experience with these applications. We are currently in talks with several CDN operators to develop a public implementation of `pcp` and offer it as a free service to improve the accuracy of replica server selection and reduce Web page

load times.

The rest of the paper is organized as follows. In Section II we discuss work related to Internet latency estimation. Section III shows the comparative evaluation of currently active latency estimation and endpoint selection tools. In Section IV we describe `pcp`. In Section V we analyze `pcp` performance against other latency approximation tools in selecting nearby hosts for clustering and selecting the nearest server. Finally, in Section VI we conclude and outline future work.

II. RELATED WORK

To motivate `pcp` and aid in interpretation of our measurements, we briefly describe existing latency estimation tools and their shortcomings.

A. ping

`ping` estimates round trip time (RTT) between network hosts with ICMP control packets. Under the assumption of path symmetry *one way* latency is often considered as RTT/2, but we will treat latency and RTT interchangeably in this paper. However, `ping` does not scale for endpoint selection in large systems [20], [21]. First, when there are many clients trying to select the closest from a few servers, each server will experience high probing traffic. For this reasons many CDN replica servers do not respond to ICMP probes. Second, to select nearby hosts in a peer-to-peer setting a client needs to probe potentially thousands of other clients – a prohibitively time consuming proposition. To combat the high cost and delay of all-to-all probing, several tools have been developed to provide latency approximation between hosts.

B. IP geolocation databases

IP geolocation databases maintain physical location entries obtained from `whois` and DNS records, or by mining websites that ask for users' physical addresses [23], [24]. Applications may compute the great-circle distance between endpoints as an estimate of network proximity. However, network latency depends on factors such as network topology, capacity, ISP peering, and traffic conditions, and so geographic proximity is often a poor predictor of latency. Additionally, when entries in geolocation databases are stale, for example when large IP blocks of CDN servers are recorded for the same location and not updated with dynamic reassignment, endpoint selection becomes inaccurate. Finally, our results show that IP geolocation databases suffer from incomplete coverage of the IP space.

C. Traceroute-based tools

Traceroute-based tools take a more dynamic approach and estimate and record the physical location of an IP address from locations of nearby routers identified by traceroute paths to that IP [26]. However, traceroute may not return information for all hops and accurate location estimates require traceroutes from multiple vantage points [35]. Paris traceroute improves the accuracy of router discovery, but has not been incorporated into the one currently active traceroute geolocation tool we were able to evaluate [26].

D. Network Coordinate Systems

To provide self-updating latency estimation without all-to-all measurements the research community has proposed several network coordinate systems (NCSs). NCSs use network measurements to build a graph of Internet topology and predict end-to-end network performance based on paths along graph edges [27], [28], [36]. Another approach is to embed nodes in a multidimensional space [29], [30], however such embeddings are sensitive to initial node placement and do not reflect Internet triangle equality violations [20]. A more direct approach based on explicit measurement of inter-AS path segments was proposed by Lee *et al.* [36].

Regardless of the underlying data structure, predictions require that a large number of path measurements be kept up to date. Systems based on the Meridian P2P measurement system reduce the problem size by maintaining measurements only between application servers and can identify the closest among them to a given IP through recursive search [37]–[39]. HybridNN takes a similar approach, but complements the search process with `ping` measurements to avoid local minima [40]. However, these tools are intended to identify the closest server to a querying node and do not provide general latency approximations, which could be used to cluster nodes in peer-to-peer applications. Finally, NCS approaches have been shown to be less accurate in edge networks and areas, where the nearest NCS landmark server might be far from clients [41].

E. Leveraging Internet infrastructure

To eliminate the needs for NCS landmark servers (with well-known coordinates) a couple of approaches leverage existing Internet infrastructure for endpoint selection. Gummadi *et al.* propose King, which approximates latency between two hosts based on the latency between their authoritative DNS servers, which are generally located near the hosts they serve [31]. To obtain latency between two DNS servers, King relies on one of the servers to allow recursive DNS queries from clients outside its subnet. Our measurements show that the number of such servers is much lower today than at the time of King's publication in 2002, since many DNS servers disable recursive queries to protect from DNS amplification attacks [42]. Turbo King (T-King) improves on the original approach by crawling the Internet to create a list of DNS servers that allow recursive queries [43]. However, even with those improvements only 54% of DNS are available for recursive queries [43].

Another tool for endpoint selection based on Internet infrastructure is CDN-based Relative network Positioning (CRP) by Su *et al.* [32]. CRP calculates the cosine similarity between CDN addresses resolved for different clients. This technique is useful in identifying clusters of nearby clients, but does not by itself predict latency within a cluster. Inability to estimate latency makes CRP unsuitable for nearest server selection without additional probing. Also, for nodes that do not share CDN servers in DNS resolutions, CRP is not able to determine proximity.

TABLE I
IP SPACE COVERAGE AND DATACENTER SELECTION ACCURACY.

	Neustar [23]	traceroute tool [26]	iPlane [27]	King [31]	T-King [43]
Coverage (%)	93.20	63.36	39.27	19.3	67.7
Accuracy (%)	26.80	22.04	9.60	N/A	N/A

F. Integration with network infrastructure

Finally, there are several proposals to improve server selection through tools that are integrated with existing network infrastructure. The Extensions to DNS (EDNS) proposal aims to improve the accuracy of DNS redirections by passing a client's IP subnet to the authoritative DNS [44]. However, EDNS mechanisms are not yet widely adopted by ISPs and open DNS servers [45]. Several solutions also propose server selection based on network measurement and server load information from cloud providers [21], [46]. Closest to our work is CloudGPS, which similarly to pcp measures network latency to a small number of nearby servers [21]. However, CloudGPS relies on active probing and AS topology information, which is difficult to keep up-to-date. Additionally, CloudGPS does not offer a solution to the client clustering problem.

III. COMPARATIVE EVALUATION OF LATENCY ESTIMATION TOOLS

To make sense of the different tools for latency estimation and endpoint selection we conducted an analysis of their coverage of the IP space and accuracy with respect to ping. We entered 100,000 random, reachable IP addresses to **currently available** endpoint selection tools to predict the closest of four public cloud datacenters in the continental US operated by different providers. Table I shows *coverage*, as the percentage of IP addresses for which each tool produced a prediction, and *accuracy* (for the addresses inside the coverage), as the percentage of time each tool selected the same datacenter as direct measurement latency using ping. We calculated Turbo King coverage from 645 million probes to random addresses performed in 2006 [43].

Our results show low accuracy across the tool categories and a high variation in coverage of the IP address space. Taken together these results motivate further work on replica selection mechanisms that combine high accuracy and broad coverage.

IV. PING THROUGH CDN PROXIES

We propose pcp – a new method for latency approximation between arbitrary hosts. pcp improves on currently available tools by achieving the following three design goals:

- **Accuracy:** pcp's latency approximation results in more accurate latency prediction and Internet endpoint selection than existing tools. pcp is also able to achieve full coverage of the IP space among participating nodes.
- **Scalability:** pcp does not require dedicated probing traffic.
- **Maintainability:** pcp does not require dedicated measurement infrastructure.

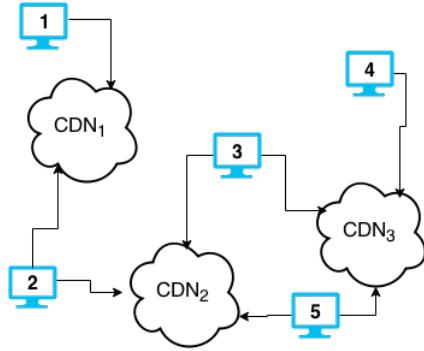


Fig. 1. Network Topology of Hosts and CDNs

A. Architecture

IP2NL estimates latency between hosts based on the shortest path between them in a virtual topology of the network built from latency measurements between clients and CDN servers. These measurements may be obtained through the passive capture of a client's Web traffic, from which pcp extracts latency from the delay of TCP OPEN requests to CDN servers. In our experiments we direct clients in Dasu and Seattle Internet testbeds to resolve and probe servers in several CDNs [33], [34].

pcp then constructs a virtual network graph, in which client and CDN nodes are connected by weighted edges representing measured latencies between them. To approximate latency between two clients in the graph pcp finds the shortest path between them and sums the edge latencies along the path.

Figure 1 demonstrates the virtual topology of a simple network containing five nodes and three CDN servers. Graph edges are client to CDN latency measurements. pcp first simplifies this graph by calculating the lowest known latency L between two CDN servers s_i and s_j as:

$$L(s_i, s_j) = \min_{c \in C_i^j} (L(s_i, c) + L(c, s_j)),$$

where L represents latency between two nodes and C_i^j is the set of clients with latency measurements to s_i and s_j . We expect (and verify in Section V) that a large set of clients with measurements to the same two CDN servers will contain a client c that accurately estimates the latency between the two servers. Clients are redirected to nearby CDNs based on their proximity in the network. By contacting an adequately sized sample of CDNs, clients will select at least a few that are located nearby in the network. Inter-server latencies may also be supplied to pcp from external sources, for example through collaboration with CDN providers.

To estimate the latency between any two clients pcp finds the shortest path between them in the virtual network graph and sums the weights of the edges along the path. For example latency between c_1 and c_4 is:

$$L(c_1, c_4) = L(c_1, s_1) + L(s_1, s_2) + L(s_2, s_3) + L(s_3, c_4).$$

When the virtual network graph is connected then there exists a path between every pair of clients. In Section V we show that a connected graph is created in practice even with

a relatively small number of nodes. As the number of nodes in the graph increases, pcp makes better latency prediction through more accurate estimates of latency between CDN servers.

pcp estimates of end-to-end latency can be used for server selection as well, when these servers collect and contribute latency measurements to CDN servers. pcp can also be used for the clustering of nearby nodes through breadth first search in the virtual network graph.

B. Feasibility

pcp integrates easily with existing Internet infrastructure. CDN servers are already widely available and well-provisioned to handle frequent requests from clients. Through passive observation of client traffic to CDNs pcp collects the latency measurements for the virtual network graph without introducing additional probing load.

Two remaining question are how large is the virtual network graph and where does pcp store it. The size of the graph for n nodes in the worst case, that is if every node shares a mutual CDN server with every other node, is equal to:

$$\binom{n}{2} \text{nodes} + \frac{\binom{n}{2}(\binom{n}{2} + 1)}{2} \text{edges.}$$

Using Dijkstra's algorithm a latency approximation from one node to every other node takes:

$$O\left(\frac{\binom{n}{2}(\binom{n}{2} + 1)}{2} + \binom{n}{2} \log \binom{n}{2}\right).$$

Our current pcp implementation uses a single server, however we are actively developing a peer-to-peer implementation of pcp measurement and latency estimation functionality, to make the storage requirement of the virtual network graph constant per client. We will make the implementation available at <http://github.com/msu-netlab/pcp>.

We believe that reliance of CDN servers, lack of dedicated probing traffic, and peer-to-peer implementation will make pcp a self-maintaining and scalable solution. To evaluate whether pcp also meets its goal of accurate latency estimation and broad IP coverage we turn to the following evaluation.

V. EVALUATION

We evaluate pcp on two different data sets provided by the Dasu and Seattle Internet testbeds [33], [34]. Seattle and Dasu operate on donated computers, servers, and smartphones. We were able to access 199 Dasu nodes and make latency measurements to 5235 distinct CDN servers. Using Seattle we collected measurements from 152 hosts to 4151 distinct CDN servers. To obtain a “ground truth” latency measurements for server selection we pinged Dasu and Seattle nodes from a set of 42 PlanetLab nodes. For ground truth in node clustering we obtained all-to-all pings among Seattle nodes. We obtained latency measurements between Dasu hosts by creating a graph of Internet routers, from inter-node traceroute measurements, and set latencies between routers as edges. We then found the network latency between every Dasu host using Dijkstra's algorithm to determine the shortest networking path

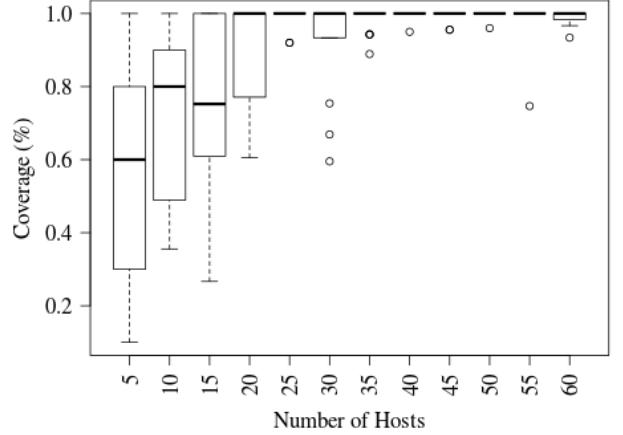


Fig. 2. Coverage of pcp for random sets of clients.

through the discovered routers. Data on both testbeds was collected during a five day period in 2014.

A. Coverage of the IP space

Our first result illustrates pcp's potential to achieve a high coverage of the IP space. We ran pcp on differently sized random subsets of clients in our datasets and calculated the percentage of clients, to which pcp was able to make latency estimates. Figure 2 shows a boxplot of our results based on 20 trials for each set size. The x-axis marks the size of the random client subsets, while the y-axis shows the pcp coverage of the set, or the percentage of reachable nodes from all other nodes in the virtual network graph. We observe that small client subsets might not have enough CDN measurements in common to form a connected virtual network graph. However, pcp's coverage rapidly improves with larger number of participating clients. With a high probability of full coverage at 25 client subset we stop the graph at 60 clients.

Thus even with a relatively small number of participating clients pcp is able to predict latency for all participating nodes with high probability. This property makes pcp suitable not only for large scale deployment as a public service, but also for inclusion in the code base of applications with a smaller user base. Of course to participate in a pcp deployment a node must report its latencies to nearby CDNs – this model has already been adopted by many NCSs systems and CRP. However, for applications such as online games, where client-side code is distributed to clients, clients that join the system are covered, while without full coverage systems like IP geolocation databases might simply not be able to estimate latency for some IPs.

B. Accuracy of latency approximations

We also evaluate the accuracy of different tools at approximating the latency between two hosts in the network. To calculate the accuracy of pcp, iPlane, and King, we subtract latency measured with ping, or traceroute from the latency approximation of each tool. Figures 3 and 4 are CDFs of accuracy for latency between each client pairing in the Dasu and Seattle datasets respectively. Correctly approximated

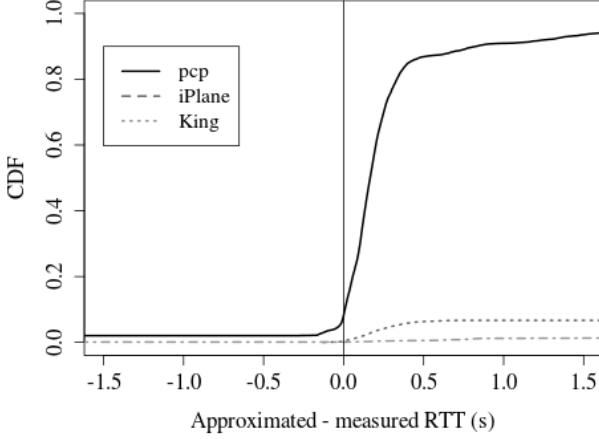


Fig. 3. Latency estimation accuracy between Dasu clients.

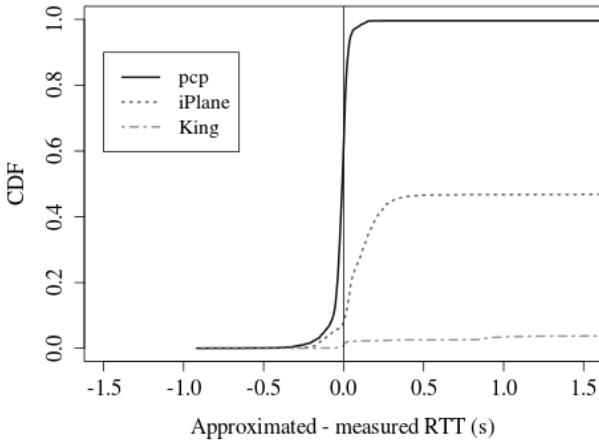


Fig. 4. Latency estimation accuracy between Seattle clients.

latencies would form a vertical line in the CDF, showing no difference from the actual measured latency. Data points to the right of the UDP ping data set are overestimates in latency while points to the left are underestimates.

`pcp` is able to approximate latency among 95% of Dasu clients to within 0.5 s and 97% of Seattle clients to within 0.2 s of direct measurement. `pcp`'s accurate latency prediction among clients allows for accurate clustering of nearby clients in peer-to-peer applications. `iPlane` and `King` are generally less accurate and achieve much lower coverage over the IPs in the Dasu and Seattle datasets. We also found that `King`'s low coverage was due to refused recursive DNS queries – `King` sent out 94,961 queries and had 76,657 of them refused. The staggering number of refusals resulted in the low coverage that is seen with `King` throughout the evaluation.

We also compare the accuracy of `pcp`, `iPlane`, and `King` in estimating latency to a set of 42 PlanetLab servers. Figures 5 and 6 are CDFs of accuracy for latency estimation between each client in the Dasu and Seattle datasets respectively and the PlanetLab servers.

`pcp` is able to approximate latency between PlanetLab servers and 96% of Dasu clients as well as 100% of Seattle clients to within 0.2 s of direct measurement. `pcp`'s accurate

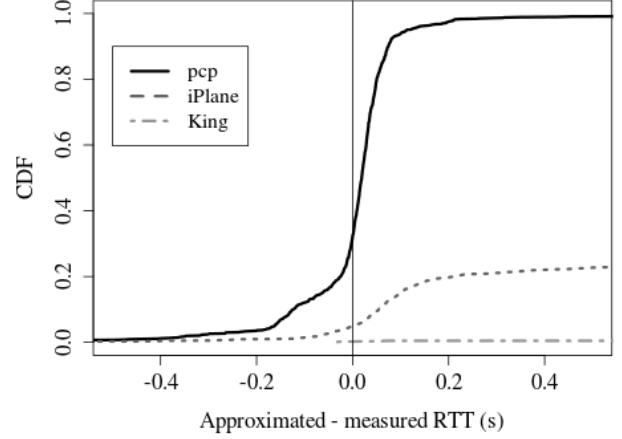


Fig. 5. Latency estimation accuracy from Dasu clients to PlanetLab servers.

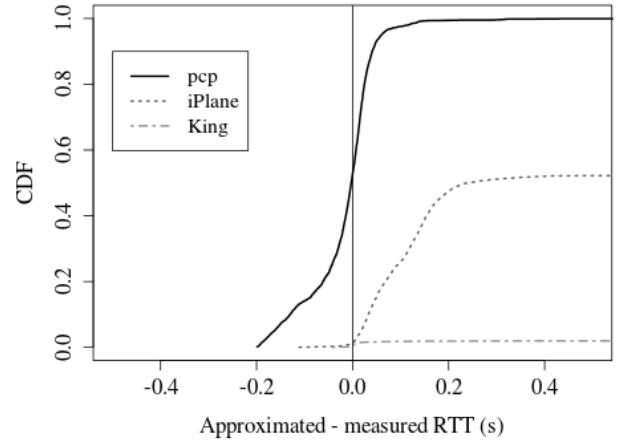


Fig. 6. Latency estimation accuracy from Seattle clients to PlanetLab servers.

latency prediction between clients and servers allows for accurate server selection in a variety of interactive Internet applications. Again, `iPlane` and `King` offer much lower coverage. The accuracy of `iPlane` is comparable to `pcp` for the clients for which it can make an estimation. The accuracy of `King` is hard to assess due to its very low coverage.

C. Accuracy of endpoint selection

Next, we wanted to understand whether `pcp`'s accuracy in latency estimation translates to more accurate selection of client clusters and servers. We define a cluster of size n to be the closest n clients to any given client c in the virtual network graph. For each client c we find the closest 10 clients as determined by latency approximations of `pcp`, `iPlane`, and `King`. We also include clusters identified based on geographic proximity from Neustar and cosine similarity of DNS resolutions from CRP. We then determine each tool's effectiveness at choosing clusters as the difference between the mean latency within clusters identified by approximation tools and direct latency measurement. We calculate this difference as:

$$\frac{\sum_{i=0}^n L_m(c, c_i)}{n} - \frac{\sum_{i=0}^n L_m(c, o_i)}{n},$$

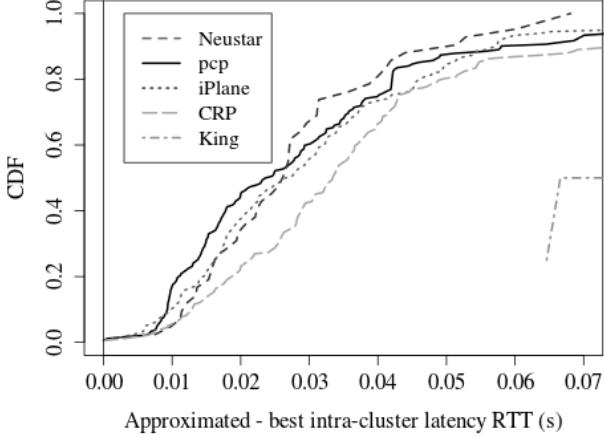


Fig. 7. Clusters of size 10 on Dasu nodes

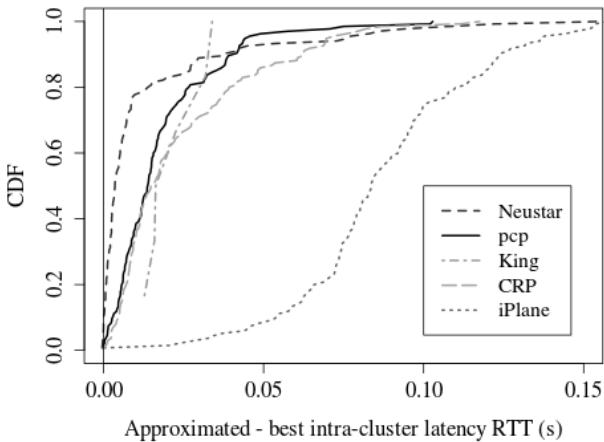


Fig. 8. Clusters of size 10 on Seattle nodes

where L_m gives the latency between two hosts measured with ICMP, $C[i..n]$ is the cluster of nodes identified by latency approximation, and $O[i..n]$ is the set of n closest nodes to c identified by ICMP latency measurement.

Figures 7 and 8 are CDFs of the difference of latency within clusters of size 10 identified by approximation tools and direct latency measurement for Dasu and Seattle clients respectively. Correctly chosen clusters would form a vertical line in the CDF, showing no difference from the clusters chosen by direct measurement. We observe that all tools select clusters with larger intra-cluster latency than direct measurement. The solid line of pcp selects clusters of comparable, or better accuracy than the other tools. For 90% of Seattle clients Neustar selects better clusters than pcp, but we observe that the difference in marginal and Neustar has lower coverage of IP space as shown Table I. Although King appears to have identified accurate clusters, it was only able to identify clusters of size 10 for 4% of nodes. Overall, pcp's accurate cluster selection allows peer-to-peer applications to achieve responsive communications among interacting peers.

Finally, we analyze the ability of each tool to accurately select the closest server to a given client. We then determine accuracy of closest server selection as the difference in latency

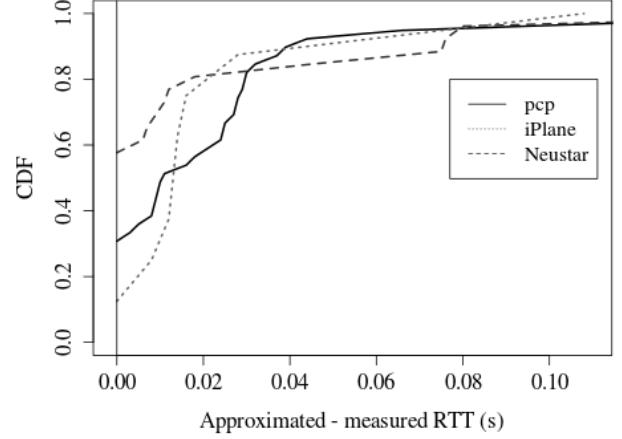


Fig. 9. Estimated server latency - real server latency on Dasu nodes

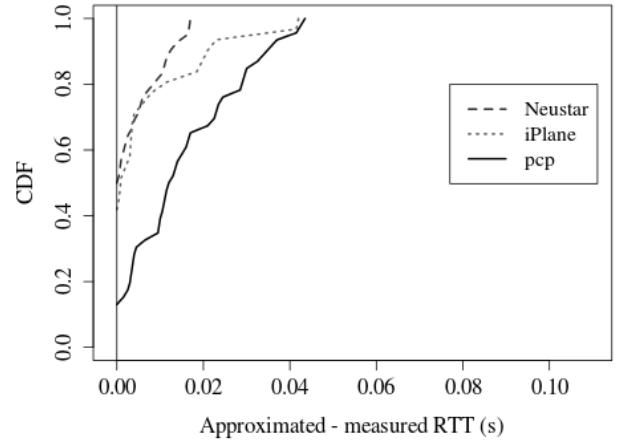


Fig. 10. Estimated server latency - real server latency on Seattle nodes

between the closest server identified by the approximation tool and direct latency measurement. Figures 9 and 10 show CDFs of the difference in latency between clients and PlanetLab servers chosen by each tool and direct measurement for Dasu and Seattle clients respectively.

In general pcp selects PlanetLab servers with error similar to Neustar and iPlane. For Seattle clients pcp's accuracy is slightly lower. Many of Seattle nodes are hosted on PlanetLab servers themselves. If a Seattle node and PlanetLab server are in the same subnet, pcp may overestimate their latency through a nearby CDN server. As part of the future work we will investigate how prevalent is client and server co-location and investigate methods for more direct latency estimation for such cases in pcp. Finally, King's low coverage prevented it from making any server selections for Dasu and only to 7 servers from Seattle clients, and so we do not consider King in these results.

VI. CONCLUSION AND FUTURE WORK

In this paper we have introduced pcp, a novel method of latency approximation between arbitrary Internet hosts. Through reliance on widespread and well-provisioned CDN infrastructure pcp offers an attractive trade-off between accuracy,

scalability, and maintainability. Through extensive evaluation on the Dasu and Seattle Internet testbeds we have shown that `pcp` achieves better IP coverage and accuracy than currently available latency estimation tools. We have also shown that `pcp` accurately selects closest Internet endpoints in server selection and client clustering tasks. We believe the `pcp`'s accuracy, scalability, and maintainability will help interactive and real-time applications provide a more responsive service to their users.

In the future we will explore a peer-to-peer implementation of `pcp` to provide an Internet-scale self-maintaining latency estimation service. A distributed implementation of a latency estimation system is critical to the longevity of a system, as well as its responsiveness during sudden surges in client request rates [21]. One of the key challenges will be to distribute and keep up-to-date the virtual network graph, such that latency query overhead and storage overhead in the peer-to-peer system remains constant. We believe such service will be useful for server selection and clustering problems in peer-to-peer systems, as well as multi-cloud application deployments as the number of publicly available cloud datacenters rapidly increases. Finally, we are currently forging partnerships with several CDN operators to obtain more accurate latency measurements between their servers, which we expect will improve `pcp`'s accuracy.

REFERENCES

- [1] E. Howard, C. Cooper, M. P. Wittie, S. Swinford, and Q. Yang, “Cascading impact of lag on user experience in multiplayer games,” in *ACM NetGames*, Dec. 2014.
- [2] N. Beighton, “The human cloud: Wearable technologies impact on society.” <http://www.rackspace.com/blog/the-human-cloud-wearable-technologies-impact-on-society/>, June 2014.
- [3] K. Ousterhout, P. Wendell, M. Zaharia, and I. Stoica, “Sparrow: distributed, low latency scheduling,” in *ACM SOSP*, Nov. 2013.
- [4] M. Abrash, “Latency the sine qua non of AR and VR.” <http://blogs.valvesoftware.com/abrash/latency-the-sine-qua-non-of-ar-and-vr/>, Dec. 2012.
- [5] O. Hansen, “The biggest problem in Augmented Reality: Latency.” <https://identifeye.wordpress.com/2013/01/03/the-biggest-problem-in-augmented-reality-latency/>, Jan. 2013.
- [6] K.-T. Chen, P. Huang, and C.-L. Lei, “Effect of network quality on player departure behavior in online games,” *Parallel Distributed Systems*, vol. 20, pp. 593–606, May 2009.
- [7] P. Tarng, K. Chen, and P. Huang, “On prophesying online gamer departure,” in *ACM NetGames*, Nov. 2009.
- [8] P. Chen and M. El Zarki, “Perceptual view inconsistency: an objective evaluation framework for online game quality of experience (QoE),” in *ACM Workshop on Network and Systems Support for Games (NetGames)*, Oct. 2011.
- [9] M. Seif El-Nasr, B. Aghabegi, D. Milam, M. Erfani, B. Lameman, H. Maygoli, and S. Mah, “Understanding and evaluating cooperative games,” in *International conference on Human factors in computing systems*, April 2010.
- [10] C. Shei and D. Williams, “Unpacking time online: Connecting Internet and massively multiplayer online game use with psychosocial well-being,” *Communication Research*, vol. 38, pp. 123–149, Feb. 2011.
- [11] R. Hirota and S. Kurabayashi, “Evaluation of fairness in multiplayer network games,” in *Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim)*, Aug. 2011.
- [12] D.-Y. Chen, H.-T. Yang, and K.-T. Chen, “Dude, the source of lags is on your computer,” in *ACM Workshop on Network and Systems Support for Games (NetGames)*, Dec. 2013.
- [13] “Delay in public games/latency issues.” <http://eu.battle.net/d3/en/forum/topic/8324862587>, Sept. 2013.
- [14] “Latency/Lag/Packet Losses. We may have found the problem..” <http://www.reddit.com/r/leagueoflegends/comments/1q85b9>, Nov. 2013.
- [15] C. Zhang, C. Huang, P. A. Chou, J. Li, S. Mehrotra, K. W. Ross, H. Chen, F. Livni, and J. Thaler, “Pangolin: speeding up concurrent messaging for cloud-based social gaming,” in *ACM CoNEXT*, Dec. 2011.
- [16] “Everything You Need To Know About CDN Load Balancing.” <http://www.webtorials.com/main/resource/papers/Dyn/paper1/CDN-LoadBalancing.pdf>, Sept 2014.
- [17] “AWS Direct Connect.” <http://aws.amazon.com/directconnect/>, Dec. 2014.
- [18] “Microsoft Azure ExpressRoute.” <http://azure.microsoft.com/en-us/services/expressroute/>, Dec. 2014.
- [19] Y. Chen, S. Jain, V. K. Adhikari, and Z.-L. Zhang, “Characterizing roles of front-end servers in end-to-end performance of dynamic content distribution,” in *ACM IMC*, Nov. 2011.
- [20] S. Agarwal and J. R. Lorch, “Matchmaking for online games and other latency-sensitive P2P systems,” in *ACM SIGCOMM*, August 2009.
- [21] C. Ding, Y. Chen, T. Xu, and X. Fu, “CloudGPS: a scalable and ISP-friendly server selection scheme in cloud computing environments,” in *Workshop on Quality of Service (IWQoS)*, June 2012.
- [22] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante, “Content delivery and the natural evolution of DNS: remote dns trends, performance issues and alternative solutions,” in *ACM IMC*, November 2012.
- [23] “Neustar — IP Intelligence.” <http://www.neustar.biz/services/ip-intelligence>, July 2014.
- [24] “IP fingerprints.” <http://www.ipfingerprints.com/>, 2013.
- [25] GeoBytes, “IP traceroute locator tool.” <http://www.geobytess.com/traceroutelocator.htm>, August 2006.
- [26] whatismyipaddress.com, “Traceroute-tool.” <http://whatismyipaddress.com/traceroute-tool>, 2013.
- [27] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, “iPlane: an information plane for distributed services,” in *Symposium on Operating Systems Design and Implementation (OSDI)*, November 2006.
- [28] 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 Transactions on Networking*, vol. 9, pp. 525–540, October 2001.
- [29] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, “Vivaldi: a decentralized network coordinate system,” in *ACM SIGCOMM*, August 2004.
- [30] J. Ledlie, P. Gardner, and M. Seltzer, “Network coordinates in the wild,” in *USENIX NSDI*, May 2007.
- [31] K. P. Gummadi, S. Saroiu, and S. D. Gribble, “King: Estimating latency between arbitrary Internet end hosts,” in *SIGCOMM Workshop on Internet Measurement*, Nov. 2002.
- [32] A.-J. Su, D. R. Choffnes, F. E. Bustamante, and A. Kuzmanovic, “Relative network positioning via CDN redirections,” in *International Conference on Distributed Computing Systems (ICDCS)*, June 2008.
- [33] M. A. Sánchez, J. S. Otto, Z. S. Bischof, D. R. Choffnes, F. E. Bustamante, B. Krishnamurthy, and W. Willinger, “Dasu: Pushing experiments to the Internet’s edge,” in *USENIX NSDI*, Apr. 2013.
- [34] J. Cappos, I. Beschastnikh, A. Krishnamurthy, and T. Anderson, “Seattle: a platform for educational cloud computing,” in *ACM SIGCSE Bulletin*, Mar 2009.
- [35] G. Connolly, A. Sachenko, and G. Markowsky, “Distributed traceroute approach to geographically locating IP devices,” in *IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems*, September 2003.
- [36] D. K. Lee, K. Jang, C. Lee, G. Iannaccone, and S. Moon, “Path stitching: Internet-wide path and delay estimation from existing measurements,” in *IEEE Infocom*, March 2010.
- [37] B. Wong, A. Slivkins, and E. G. Sirer, “Meridian: a lightweight network location service without virtual coordinates,” in *ACM SIGCOMM*, August 2005.
- [38] M. J. Freedman, K. Lakshminarayanan, and D. Mazières, “OASIS: anycast for any service,” in *USENIX NSDI*, May 2006.
- [39] B. Wong and E. G. Sirer, “ClosestNode.com: an open access, scalable, shared geocast service for distributed systems,” *SIGOPS Operating Systems Review*, vol. 40, pp. 62–64, January 2006.
- [40] Y. Fu, Y. Wang, and E. Biersack, “HybridNN: an accurate and scalable network location service based on the inframetric model,” *Future Generation Computer Systems*, vol. 29, pp. 1485 – 1504, Aug. 2013. Including Special sections: High Performance Computing in the Cloud & Resource Discovery Mechanisms for {P2P} Systems.
- [41] tech. rep.
- [42] K. McCaney, “Universities get schooled on DNS amplification attacks.” <http://gcn.com/articles/2013/05/13/universities-get-schooled-dns-amplification-attacks.aspx>, May 2013.
- [43] D. Leonard and D. Loguinov, “Turbo King: framework for large-scale Internet delay measurements,” in *IEEE Infocom*, pp. –, Apr. 2008.

- [44] S. Souders, “Extension Mechanisms for DNS (EDNS(0)).” <http://tools.ietf.org/html/rfc6891>, Apr 2013.
- [45] J. P. Rula and F. E. Bustamante, “Behind the Curtain: Cellular DNS and Content Replica Selection,” in *Proceedings of the 2014 Conference on Internet Measurement Conference*, pp. 59–72, ACM, 2014.
- [46] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, “DONAR: decentralized server selection for cloud services,” in *ACM SIGCOMM*, August 2010.