This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

Traceroute-Based Topology Inference without Network Coordinate Estimation

Xing Jin*, Wanqing Tu† and S.-H. Gary Chan*

* Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Email: {csvenus, gchan}@cse.ust.hk

† Department of Computer Science
University College Cork
Cork, Ireland
Email: wanqing.tu@cs.ucc.ie

***Abstract*— Underlay topology information is important to construct efficient overlay networks. To achieve end-to-end network topology inference among a group of hosts, traceroute-like tools are often used. Previously, Max-Delta has been proposed to infer a highly accurate topology with a low number of traceroutes. However, Max-Delta relies on external tools to estimate host coordinates, which incur considerable deployment overhead. In this paper, we consider novel inference schemes with no coordinate estimation. One choice is to select long paths to traceroute. That is, based on existing traceroute results, each host can estimate the distance between another host and itself. It can then select the host with the largest distance between them as the traceroute target. We call this scheme *Longest-Path-First (LPF)*. Similarly, we can define *Shortest-Path-First (SPF)* inference.**

The intuition may indicate that LPF performs better than SPF, as longer paths often contain more underlay links and routers, and hence more undiscovered links or routers. However, our simulation results on Internet-like topologies show that SPF can achieve comparable performance with Max-Delta, while LPF performs even worse than a random inference scheme. To explain the results, we analyze the statistics of all-pairs paths between hosts. Our results show that long paths have serious overlaps on underlay links, showing much higher path stress than short paths. We also find that there exist quite a few links only appearing in short paths and seldom appearing in long paths. Therefore, with the same number of traceroutes, SPF can discover more underlay links and routers than LPF. Furthermore, as SPF prefers short paths, a traceroute in SPF sends less probing packets and consumes less network resource than that in LPF. Therefore, SPF is a highly efficient inference scheme with low deployment overhead and low measurement overhead.

I. Introduction

In the recent years, overlay networks have been increasingly used to deploy network services. Examples include overlay path routing, application-layer multicast (ALM), peer-to-peer streaming and file sharing, and so on [1]–[4]. In order to build an efficient overlay network, the knowledge of the underlay topology is important. For example, two seemingly disjoint overlay paths may share common underlay links; therefore the selection of overlay paths without the knowledge of underlay may lead to serious link congestion. However, it is not trivial to obtain an underlay topology through end-to-end measurements. Border Gateway Protocol (BGP) routing tables can construct an autonomous-system-level topology [5]. But the tables are stored at specific gateway routers and are not publicly available. Techniques like network tomography infer a topology based on correlation between path properties [6]. Although it is an end-to-end approach, the resultant topology is often inaccurate and unstable. Therefore, the most commonly used tool is traceroute, which can explicitly extract the router-level path between a pair of hosts [7]. In fact, it has been shown that ALM based on router-level topology information can achieve substantially low end-to-end delay, low physical link stress and high tree bandwidth [8], [9].

Max-Delta has been proposed to efficiently infer the underlay topology among a group of hosts based on traceroutes [10]. In Max-Delta, hosts first use tools like GNP [11] or Vivaldi [12] to estimate their coordinates. They then report the coordinates to a central server. The server chooses the best set of paths for hosts to traceroute. Clearly, coordinate estimation by external tools leads to considerable implementation and deployment overhead. For example, if GNP is used, around $15 - 20$ public landmarks need to be deployed over the network. These landmarks will receive ping requests from all the hosts in the system, and should have enough edge bandwidth and computational power. In order to reduce estimation error, it is better to deploy these landmarks in different areas around the network. This further increases the difficulty in deployment. Similarly, other tools have their own deployment overhead. Furthermore, many of these tools do not provide public source codes. It takes additional effort to implement the estimation algorithms. Therefore, in this paper we consider novel inference schemes with no coordinate estimation, which can be more easily implemented and deployed than Max-Delta.

We have several ways to select traceroute paths. One choice is to preferentially select long paths to traceroute. Based on existing traceroute results, the distance between any two hosts can be estimated. A host can then select the longest unmeasured path as its traceroute target. We call this method *Longest-Path-First (LPF)*. An opposite of LPF is *Shortest-Path-First (SPF)*, where a host selects the shortest unmeasured path as its traceroute target. Intuitively, people may feel

This work was supported, in part, by Direct Allocation Grant at the HKUST (DAG05/06.EG10), and Hong Kong Research Grant Council (611107).

978-1-4244-2075-9/08/$25.00 ©2008 IEEE

that LPF is more reasonable and should perform better than SPF, because a longer path may contain more undiscovered links and routers. However, our simulations on Internet-like topologies show the opposite results. While SPF can achieve comparable performance with Max-Delta, LPF performs much worse than them. Even a random inference scheme performs better than LPF.

To explain the results, we analyze the statistics of all-pairs paths between hosts. Our analysis shows that long paths have serious overlaps on underlay links, showing much higher path stress than short paths. From the all-pairs paths between 500 hosts, short paths with link hops no more than 5 cover over 95% links, but long paths with link hops no less than 6 only cover around 79% links. It shows that there exist quite a few links only appearing in short paths and seldom appearing in long paths. Therefore, with the same number of traceroutes, SPF can discover more underlay links and routers than LPF. Furthermore, as SPF prefers short paths, a traceroute in SPF sends less probing packets and consumes less network resource than that in LPF. Therefore, SPF is a highly efficient inference scheme with low deployment overhead and low measurement overhead.

We briefly review related work as follows. Traceroute-like tools have been widely used in Internet measurements such as Skitter, Mercator and Rocketfuel [13]–[15]. Skitter sends traceroute packets from different locations worldwide to actively measure the Internet topology. Mercator utilizes a modified version of traceroute to reduce probing time. Rocketfuel combines information from BGP tables, traceroutes and DNS to infer ISP topologies. All these works focus on Internet- or ISP-level topology inference and the major concern is how to discover a complete network topology including all the routers and links. However, in our study we are only interested in the topology among a certain group of hosts that are arbitrarily distributed in the Internet. Furthermore, we only need a highly accurate topology, because most overlay applications are tolerant to small distortion of the underlay topology. The key problem is hence how to reduce measurement cost. Similar to Max-Delta, in this paper we use a central server for result collection and path selection. Using the method in [16], we can develop a fully distributed version for our scheme. We may also integrate the Doubletree algorithm [17] into our scheme to reduce measurement redundancy. Doubletree modifies the working process of traceroute in order to avoid repeatedly discovering the same links or routers by multiple traceroutes. The integration details are similar to [18].

In this paper we assume that the traceroute path from a host A to another host B is the reverse of the path from B to A. The rest of the paper is organized as follows. In Section II, we describe LPF and SPF inference schemes. In Section III, we compare the performance of different inference schemes through simulations. We also analyze path properties to explain why SPF is better than LPF. We finally conclude in Section IV.

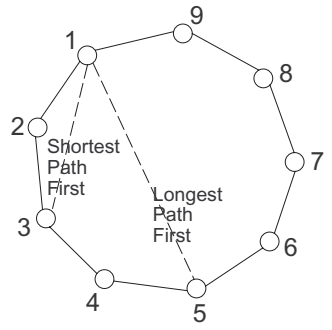

Fig. 1. An example of target selection in topology inference.

II. Inference With No Coordinate Estimation

Max-Delta needs to use external tools for coordinate estimation. These tools introduce additional implementation and deployment overhead. We hence consider a simpler inference scheme with no coordinate estimation.

The architecture of the inference scheme is described as follows. Suppose that there are N hosts in the system. Each host has a unique host ID, which is an integer between 1 and N. Similar to Max-Delta in [10], we consider that a central server collects all useful information from hosts and selects traceroute paths for hosts. The inference procedure is divided into multiple iterations. In each iteration, the server selects a target for each host to traceroute. Hosts then traceroute their targets and report the results to the server. The server combines all the results obtained in the iteration and based on that, starts the next iteration on target assignment. Such process is repeated until a certain stop rule is achieved (e.g., reaching a certain number of iterations). In the first iteration, we require the host with ID i to traceroute the host with ID $(i \ mod \ N) + 1$. That is, host 1 traceroutes host 2, host 2 traceroutes host 3, and so on. Figure 1 shows an example of first-iteration traceroutes, where $N = 9$. Solid lines in the figure indicate the overlay paths that have been tracerouted. Note that we assume paths are symmetric on the router level. After the first-iteration traceroutes, the hosts form a connected circle on the overlay.

We have several ways for target selection in the following iterations. The first method is to select the longest unmeasured path on the partially discovered topology. Based on available traceroute results, we can build a partially discovered topology. As the topology is connected (by following the above first-iteration traceroutes), we can use shortest path routing to compute the distance between any two hosts. Denote the distance on the partially discovered topology between two hosts A and B as $D_p(A, B)$. If the path between A and B has been tracerouted, $D_p(A, B)$ can be computed as half of the round-trip time between A and B. Given a certain host A, the server computes D_p values for all unmeasured paths between A and other hosts. The server then selects the path with the maximum D_p value as A's traceroute target. We call the method *Longest-Path-First (LPF)*.

We show an example how the server selects a traceroute target for host 1 in Fig. 1. The server computes the D_p values between 1 and other hosts in the circle-like topology. For example, $D_p(1,5)$ is equal to the end-to-end distance of path $1-2-3-4-5$. Suppose that

$$D_p(1,5) = \max_{i\in[3,8]} D_p(1,i).$$

Here paths $1-2$ and $1-9$ have been tracerouted and will not be tracerouted again. The server then selects host 5 as the traceroute target for host 1. The intuition behind LPF is that a long path in the network usually contains many hops and routers. Therefore, among all the unmeasured paths, a longer one may contain more undiscovered links and routers.

Another selection method is the opposite of LPF. Each time, the server selects from all the candidate paths the one with the minimum D_p value as the next traceroute target. We call it *Shortest-Path-First (SPF)*. For example, in Fig. 1, host 3 is the closest to host 1 on the partially discovered topology among hosts $3-8$. The server then selects host 3 as the traceroute target for host 1.

LPF seems more reasonable and more efficient than SPF. We expect that a long path contains more undiscovered links and routers than a short path. However, our simulation results on Internet-like topologies completely break such expectation. In our simulations, SPF can quickly infer a highly accurate topology within a few iterations. LPF, on the contrary, cannot infer an accurate topology even after a significant number of iterations. We will show and explain the results in the following.

III. Performance Evaluation and Analysis

In this section, we first show the performance of LPF and SPF, and then analyze why SPF performs better than LPF.

A. Performance of Inference Schemes

We generate a number of *Transit-Stub* topologies with GT-ITM [19]. Each topology is a two-layer hierarchy of transit networks and stub networks. A topology contains $6,000$ routers and around $42,000$ links. We randomly put $N = 500$ hosts into the network. Each host is connected to a unique stub router with 1ms delay, while the delay of core links is given by the topology generator. We use the following metrics to evaluate an inference scheme.

- *Link ratio*: defined as the ratio of the number of links in the inferred topology to the total number of links in the actual underlay topology [10].
- *Router ratio*: defined as the ratio of the number of routers in the inferred topology to the total number of routers in the actual underlay topology [10].
- *Resource usage*: Given a traffic between two points, the resource usage is computed as the size of traffic packets times the delay between the two points. We compute the resource usage of an inference scheme as the total resource usage for traceroute measurements in the inference. The resource usage of a single traceroute can be computed as in [20].
- *Number of Internet Control Message Protocol (ICMP) messages*: defined as the number of ICMP requesting messages sent by traceroutes in the inference. As in [20], we assume that a traceroute sends three ICMP requesting messages for each TTL value.

We also implement a *Random-Path (RP)* inference scheme and use its results as the performance benchmark. In RP, each host randomly selects an unmeasured path as its target in an inference iteration.

Figure 2 compares the performance of different inference schemes. Figure 2(a) shows the link ratios achieved by the schemes in different iterations. Max-Delta is the quickest to achieve 95% link ratio, only after 4 iterations. SPF is slightly slower, after 6 iterations. After 15 iterations, SPF achieves slightly higher link ratio than Max-Delta, and both of them can discover over 98% links. On the other hand, LPF performs the worst among the schemes. In the first 30 iterations, its link ratio increases from 71.4% to 74.4%. This is even worse than RP. Figure 2(b) shows the router ratios achieved by the schemes in different iterations. SPF and Max-Delta can discover over 99% routers after 4 iterations. SPF is slightly better than Max-Delta. On the contrary, LPF and RP perform much worse. They can only discover around 93% routers after 30 iterations. And their curve trends show that increasing the number of iterations does not help much. From these two figures, we can see that SPF achieves comparable performance with Max-Delta. Both of them can quickly discover a highly accurate topology. LPF is not efficient in discovering links or routers. It is even worse than RP.

Figure 2(c) shows the accumulative resource usage achieved by the schemes. As expected, LPF consumes the most network resource among the schemes. As it always selects long paths to traceroute, its traceroute averagely sends more probing packets than that of other schemes, and these packets traverse longer distances. On the contrary, SPF preferentially selects short paths. It hence achieves the lowest resource usage. Max-Delta achieves similar resource usage as RP. Their resource usage is between that of LPF and SPF. As shown, after 30 iterations, the resource usage of SPF is only 37.3% of that of Max-Delta, 36.2% of that of RP, and 20.9% of that of LPF. Therefore, during the inference procedure, SPF generates much less traffic and consumes much less network resource than other schemes. It hence significantly reduces the measurement overhead. Figure 2(d) shows the number of ICMP requesting messages sent by the schemes. As LPF prefers long paths consisting of many routers, it sends the most ICMP messages. As a comparison, SPF sends the least ICMP messages. This confirms the results in Figure 2(c).

B. Why Shortest-Path-First Better?

We now explore the reason for SPF's good performance and LPF's bad performance. Given $N = 500$, all-pairs traceroutes between hosts result in $N(N-1)/2 = 124,750$ paths (assuming paths are symmetric). We analyze the properties of the paths in a typical simulation. In the simulation, there

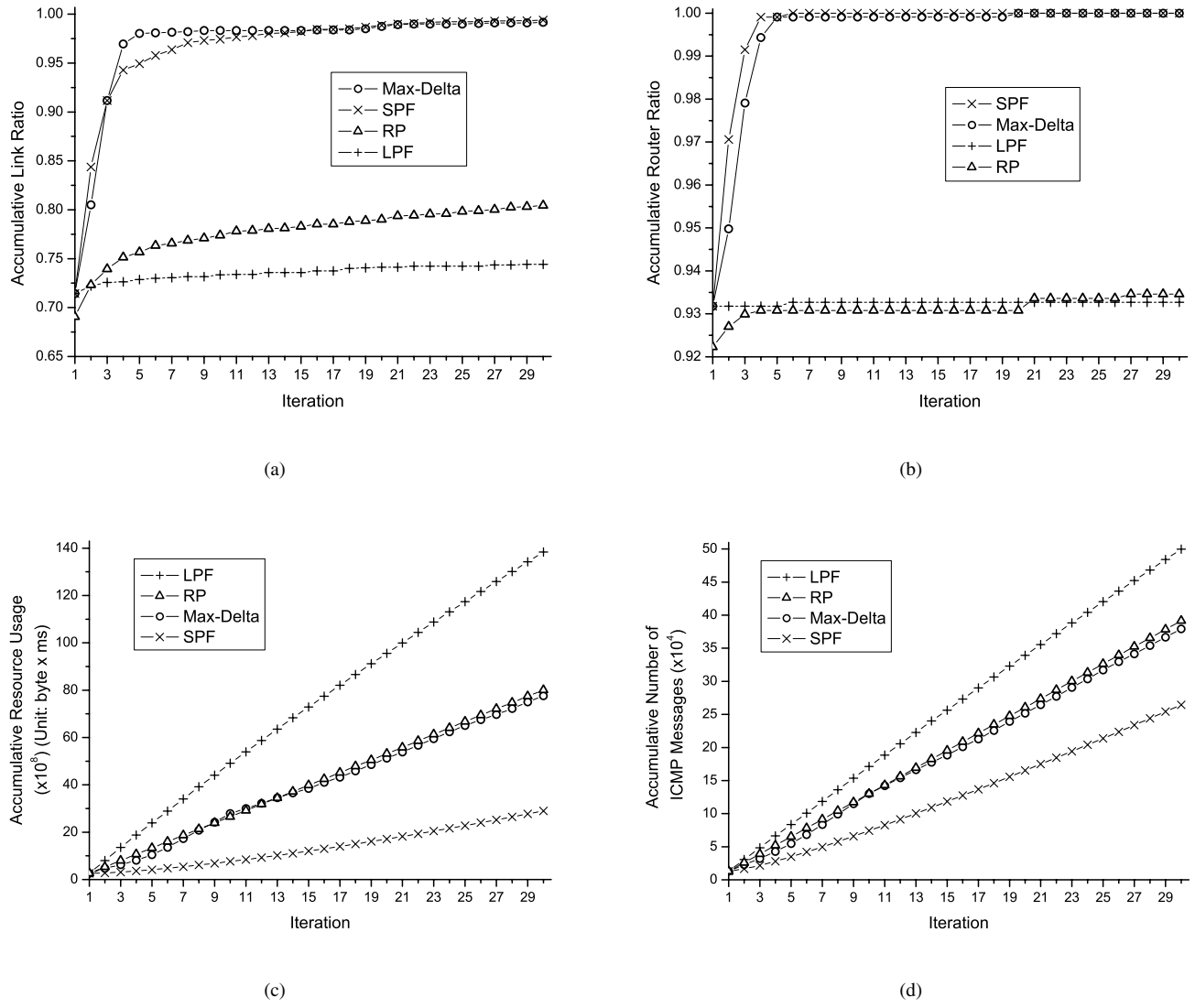

Fig. 2. Performance comparison of different inference schemes. (a) Accumulative link ratio. (b) Accumulative router ratio. (c) Accumulative resource usage. (d) Accumulative number of ICMP messages.

are in total $1,677$ different links and $1,055$ different routers in the paths.

Given a set of traceroute paths, define the *stress* of a link as the number of paths that cross the link. Based on that, we define three types of *path stress*. The maximum stress of a path is defined as the maximum stress of links in the path. The minimum stress of a path is defined as the minimum stress of links in the path. The average stress of a path is defined as the average stress of links in the path. Figure 3 shows the path stress versus the path length (in terms of hops), and the number of paths with different length. In our simulation, the maximum and minimum path length is 17 and 1, respectively. When the path length increases from 1, the number of paths increases. The number of paths reaches the top when the path length is 9. It then decreases when the path length continues increasing. As shown, most paths (around 93.3%) have length between 5 and 12. From the figure, we can see that path stress (the maximum, or minimum, or average one) generally increases with path length. For example, when path length is 4, the average path stress is 1468.6. When path length is 15, the average path stress is 3463.9. Therefore, a long path has much higher average path stress than a short path. In other words, on average, a link in a short path is crossed by less paths than a link in a long path. The results show that long paths have serious overlaps over underlay links. It is hence not efficient to traceroute long paths to discover new links.

Figure 4 shows the accumulative link ratio achieved by paths no more than certain length. For example, path length 5 corresponds to 95.9% in the curve. It means that if we only measure paths whose length is less than or equal to 5, we can discover 95.9% links on the underlay. When path length is 10, such value is 100%. It means that we can cover the whole

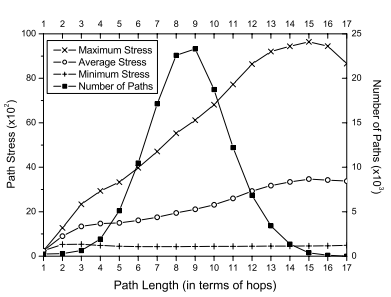

Fig. 3. Path stress and distribution with different path length.

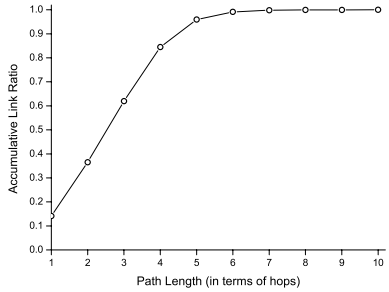

Fig. 4. Accumulative link ratio by paths no more than certain length.

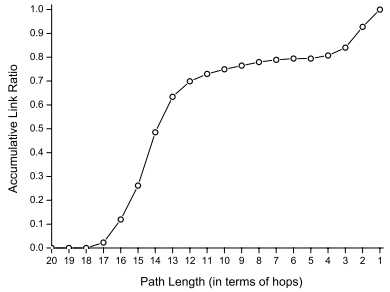

Fig. 5. Accumulative link ratio by paths no less than certain length.

underlay topology using paths whose length is no more than 10. The figure shows that short paths have contained most of underlay links. We can discover almost all links by only tracerouting short paths.

Figure 5 shows the accumulative link ratio achieved by paths no less than certain length. For example, path length 6 corresponds to 79.4% in the curve. It means that if we only measure paths whose length is more than or equal to 6, we can discover 79.4% links on the underlay. Such value reaches 100% only when path length is 1. This curve explains why LPF performs badly. In the underlay topology, some links only appear in short paths. For example, as the curve shows, 20.6% links do not appear in paths with length more than or equal to 6. As a result, if we preferentially select long paths to traceroute as in LPF, we cannot discover these links until the end of the inference procedure (when all the paths have been tracerouted).

In summary, long paths are seriously overlapped. It is difficult to discover new links by tracerouting long paths. And there exist quite a few links only appearing in short paths but not in long paths. Only tracerouting long path cannot discover these links. On the contrary, almost all links in long paths appear in short paths. We can discover most of the links by only tracerouting short paths.

IV. Conclusion

In this paper, we study traceroute-based inference schemes with no need of coordinate estimation. We propose LPF and SPF, and evaluate them through simulations. Our results show that SPF can achieve comparable performance with Max-Delta, a highly efficient inference scheme with coordinate estimation. But LPF performs even worse than a random inference scheme. We analyze path properties to explain the reason for the results. Our study indicates that preferentially tracerouting short paths in topology inference can quickly discover underlay links. In the future, we will validate our findings on real Internet topologies.

References

[1] Y. H. Chu, S. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, pp. 1456–1471, Oct. 2002.

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. ACM SIGCOMM'02*, Aug. 2002, pp. 205–217.

[3] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: A public DHT service and its uses," in *Proc. ACM SIGCOMM'05*, Aug. 2005, pp. 73–84.

[4] Y. Chawathe, S. Ramabhadran, S. Ratnasamy, A. LaMarca, J. Hellerstein, and S. Shenker, "A case study in building layered DHT applications," in *Proc. ACM SIGCOMM'05*, Aug. 2005, pp. 97–108.

[5] F. Wang and L. Gao, "On inferring and characterizing Internet routing policies," in *Proc. Internet Measurement Conference'03*, Oct. 2003, pp. 15–26.

[6] M. Coates, R. Castro, R. Nowak, M. Gadhiok, R. King, and Y. Tsang, "Maximum likelihood network topology identification from edge-based unicast measurements," in *Proc. ACM SIGMETRICS'02*, 2002, pp. 11–20.

[7] Traceroute. [Online]. Available: http://www.traceroute.org/

[8] M. Kwon and S. Fahmy, "Topology-aware overlay networks for group communication," in *Proc. ACM NOSSDAV'02*, May 2002, pp. 127–136.

[9] X. Jin, Y. Wang, and S.-H. G. Chan, "Fast overlay tree based on efficient end-to-end measurements," in *Proc. IEEE ICC'05*, May 2005, pp. 1319–1323.

[10] X. Jin, W.-P. K. Yiu, S.-H. G. Chan, and Y. Wang, "Network topology inference based on end-to-end measurements," *IEEE J. Select. Areas Commun.*, vol. 24, no. 12, pp. 2182–2195, Dec. 2006.

[11] T. S. E. Ng and H. Zhang, "Predicting Internet network distance with coordinates-based approaches," in *Proc. IEEE INFOCOM'02*, June 2002, pp. 170–179.

[12] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Proc. ACM SIGCOMM'04*, Aug. 2004, pp. 15–26.

[13] Skitter. [Online]. Available: http://www.caida.org/tools/measurement/skitter/

[14] R. Govindan and H. Tangmunarunkit, "Heuristics for Internet map discovery," in *Proc. IEEE INFOCOM'00*, March 2000, pp. 1371–1380.

[15] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. ACM SIGCOMM'02*, Aug. 2002, pp. 133–145.

[16] X. Jin, Q. Xia, and S.-H. G. Chan, "A distributed approach to end-to-end network topology inference," in *Proc. IEEE ICC'07*, June 2007, pp. 1704–1709.

[17] B. Donnet, P. Raoult, T. Friedman, and M. Crovella, "Efficient algorithms for large-scale topology discovery," in *Proc. ACM SIGMETRICS'05*, June 2005, pp. 327–338.

[18] X. Jin, W.-P. K. Yiu, and S.-H. G. Chan, "Improving the efficiency of end-to-end network topology inference," in *Proc. IEEE ICC'07*, June 2007, pp. 6454–6459.

[19] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. IEEE INFOCOM'96*, March 1996, pp. 594–602.

[20] X. Jin, Q. Xia, and S.-H. G. Chan, "A cost-based evaluation of end-to-end network measurements in overlay multicast," in *Proc. IEEE INFOCOM MiniSymposium'07*, June 2007, pp. 2581–2585.