A method for placing traceroute-like topology discovery instrumentation

Wei Han
State Key Lab. of Software Develop Environment
Beihang University
Beijing, China
hanwei@nlsde.buaa.edu.cn

Ke Xu
State Key Lab. of Software Develop Environment
Beihang University
Beijing, China
kexu@nlsde.buaa.edu.cn

Abstract—**An accurate map of the Internet is very important for studying the network's internal structure and network management. The main approach to map the Internet is to collect information from a set of sources by using traceroute-like probes. In a typical mapping project, active measurement sources are relatively scarce while traceroute destinations are plentiful, which makes the sampled graph quite different from the original one. So, it becomes very important to determine how to place these sources such that the sampled graph can be closer to the original one, especially in the case that the number of sources is limited. In this paper, we investigate the relationship between the placement of traceroute sources and their sampled result, which, to our knowledge, has not been systematically studied before. Based on the relationship, we propose a method on how to place the traceroute sources. We show that the graph sampled from sources selected by our method is more accurate than the ones randomly selected. We also validate our conclusion using the raw trace data of skitter project.**

Keywords- topology discovery; traceroute sources; placement

I. INTRODUCTION

A highly accurate map of the Internet topology is a prerequisite to model, analyze and test the Internet. It is also very important for studying the network's internal structure and network management. Now, the main approach to map the Internet is to use traceroute, which can report the interfaces along the IP path from a source to a destination. The topology graph can be obtained by merging the traceroute results of each source. Traceroute-like sampling of the Internet has been widely used in a lot of topology discovery systems [2, 3, 4, 5, 6].

Traceroute sources require deployment of dedicated measurement infrastructure, so they are always very scarce compared with the destinations. For example, skitter[1], a very famous project of topology discovery and analysis , sends traceroute probes from 25 sources deployed all over the world to more than 971,000 destinations. Due to the limitation of the number of sources, the graph obtained by the method can be considerably different from the original one[6]. Lakhina et al.[7] find that traceroute probes are more likely to find nodes and links very close to the source. Shavitt and Shir[8] analyze the result of DIMES[9] and show that by adding traceroute sources placed at the periphery of the network, DIMES can find many peering links between small ISPs, while these links can hardly be found by other projects with limited sources.

While more traceroute sources are needed in order to get an accurate topology graph, deployment of the instrumentation can be quite costly and more sources would result in sending excessive redundant probes into the Internet[10], which may affect the usual use of Internet. Therefore, when the number of sources is fixed, determining how to place these sources so that the sampled graphs can be closer to the original ones becomes a critical problem. Most topology discovery projects only consider the factor of geography when placing their sources and make them geographically distributed in the Internet. The main contribution of this paper is a method to place the traceroute sources based on the analysis of the features of traceroute. We show that the graph sampled from sources selected using our method is more accurate than the ones randomly selected. We also validate our method using the raw trace data of skitter[1].

The rest of the paper is organized as follows. First, we present some related work in section II. In section III, we establish some basic definitions. In section IV, we analyze the features of traceroute-like probes. Based on this, we derive the relationship between the placement of the traceroute sources and the graphs sampled from the sources in section V, and validate the relationship using both the trace data of a simulated network and the raw data of skitter[1]. Then, we propose a method on the placement of traceroute sources in section VI. Finally, we summarize, conclude and discuss future work in section VII.

II. RELATED WORK

Barford et al.[11] study the marginal utility of adding traceroute sources and destinations. They find that the marginal utility of adding traceroute sources beyond the second source diminishes quickly. They also argue that the diminishing marginal utility does not imply that the overall coverage obtained is high.

Dall'Asta et al.[12] show that the probability that a node or an edge can be detected by traceroute probes depends on the betweenness centrality of the element and the density of

This work has been supported by National 973 Program of China (Grant No. 2005CB321901) and Beijing Nova Program (Grant No. 2005B12).

traceroute sources and destinations. They recommend that sources should be placed on the low-connectivity nodes because of the correlation between connectivity and betweenness. However, this method is not verified by any experiment. In section VI, we will perform an experiment with this method and compare it with the method we propose.

III. DEFINITIONS

The Internet topology can be naturally modeled as an undirected graph $G=(V, E)$, where V denotes the set of vertexes (nodes) and E is the set of edges (links). The sampled graph induced by source s is a subgraph of G and we will use $G_S=(V_S, E_S)$ to denote it. Since this paper focuses on the placement of traceroute sources, we always need to compare the subgraphs induced by different sources. Given two sources $s1$ and $s2$ with the subgraphs G_{S1} and G_{S2} induced by each one, we define the intersection and union of the two subgraphs as follows:

Definition 1:

$$G_{S1}\bigcap G_{S2}=((V_{S1}\bigcap V_{S2}),(E_{S1}\bigcap E_{S2}))$$
$$G_{S1}\bigcup G_{S2}=((V_{S1}\bigcup V_{S2}),(E_{S1}\bigcup E_{S2}))$$

Breadth-first search (BFS) is a simple algorithm to explore a graph, in which we start exploring from a node s in all possible directions, adding nodes one "layer" at a time. We will use $L(s,v)$ to denote the layer in which v is explored in the BFS started from s.

Definition 2:

$$L(s,v)=\begin{cases} 0, \text{if } (v=s) \\ L(s,v')+1, \text{if } (v',v)\in E, v' \text{ is explored before } v \end{cases}$$

Using above definitions, we can define the number of nodes on each layer. Let $N(s,l)$ denote the number of nodes on layer l.

Definition 3:

$$N(s,l)=\left|\{v|v\in V, L(s,v)=l\}\right|$$

We use $LMN(s)$ to denote the layer that contains the maximum number of nodes, i.e. the layer l that maximize $N(s,l)$.

IV. THE FEATURES OF TRACEROUTE PROBES

In order to investigate the traceroute-like exploration process, we generate a graph based on the EBA model[13]. Then we analyze the features of traceroute and classify the nodes and edges that cannot be detected by traceroute probes into two categories.

A. *The graph used for simulation*

A graph whose topological properties are close to the Internet is needed in order to simulate the traceroute probes. Faloutsos et al.[14] propose the power-law relationship of the Internet topology, which is widely believed to be the most important feature of many complex networks. Based on power-law relationship, a lot of topology models[13,15,16,17] have been proposed. We will choose the EBA[13] model to generate the graph for our simulation and use the typical parameters of the EBA model.

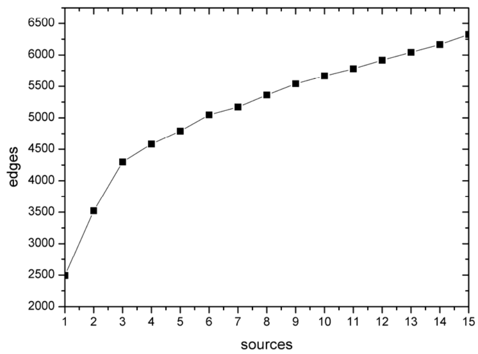


Figure 1. Number of edges detected as sources are added

The routing policy has to be decided to simulate the traceroute process. In the real Internet, there are many applied routing protocols, e.g. BGP and OSPF. The principle of routing protocols is to make packets in the network reach their destinations as soon as possible, but the actual routing path can be different from the shortest path due to commercial and political factors. Despite these factors, a reasonable approximation of the route traversed by traceroute-like probes is the shortest path between the source and the destination[12]. In the case where there are equivalent shortest paths between two nodes, Dall'Asta et al.[12] define three routing selection mechanisms: USP(Unique Shortest Path), RSP(Random Shortest Path) and ASP(All Shortest Path). Actual traceroute probes may contain a mixture of these three mechanisms. However, we choose USP policy for our simulation because the USP procedure is the closest to the one time running of traceroute probes and represents the worst case scenario.

B. *Marginal utility of adding traceroute sources*

It is shown in [11] that the marginal utility diminishes quickly when sources are added in a traceroute-like process. The authors also point out that the diminishing marginal utility does not imply that the coverage of nodes and edges is high. We investigate this by simulation on the graph generated in the previous subsection. Fig. 1 shows the results of the marginal utility of adding sources. When there are 15 sources and all nodes are destinations, the edge coverage is 57.5%. This verifies the conclusion that the overall coverage can still be low even if the marginal utility diminishes quickly. We also see that the edge coverage of a single source exploration is only 22.7%, i.e. 77.3% of the edges are invisible from a single measurement source even if the destinations are plentiful. In the following subsections, we will investigate the invisible part of the graph and find out the reasons for the low coverage.

1) *Cross-link*

The subgraph induced by a single source can be regarded as a spanning tree rooted at the source. If all the traceroute probes are started simultaneously, the traceroute process can

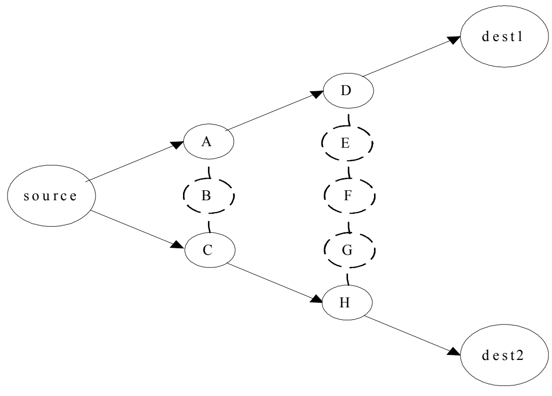


Figure 2. Illustration of cross-link

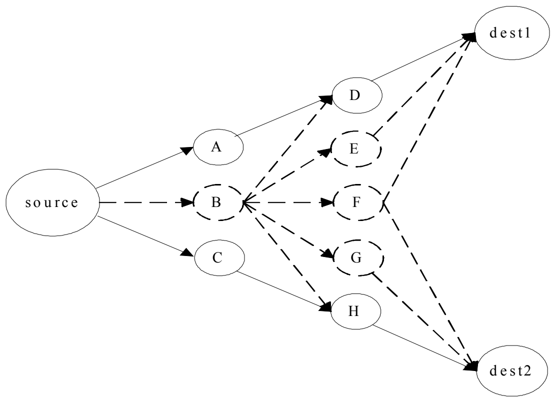


Figure 3. Illustration of traceroute probes with equivalent shortest paths. Nodes and edges with dashed lines can not be detected in this procedure.

be abstracted as the breadth-first search of the spanning tree and "layer" in the breadth-first search corresponds to "hop" in traceroute. Cross-link[18] is a link joining two nodes of the same layer, and thus cannot be detected by the breadth-first traverse of the spanning tree. For example, in fig. 2, the nodes and edges connecting *H* and *D* cannot be detected by traceroute from *source* to *dest1* and *dest2* because *H* and *D* are on the same layer of the spanning tree.

2) Equivalent shortest paths

Besides cross-link, nodes and edges may also be undetected by traceroute probes for the existence of equivalent shortest paths. This is illustrated in fig. 3, where there are 4 equivalent shortest paths from *source* to *dest1*, but only one of them can be detected in one time exploration of traceroute.

V. THE RELATIONSHIP BETWEEN PLACEMENT OF TRACEROUTE SOURCES AND THEIR SAMPLED RESULT

As mentioned in the previous section, the subgraph induced by a single source is actually a spanning tree rooted at the source. Lakhina et. al.[7] shows that there is not much difference among the coverage of the subgraphs induced by each source. So, if the number of sources is fixed, the coverage of the merged graph actually lies on the size of the intersection of every pairs of the subgraphs. For example,

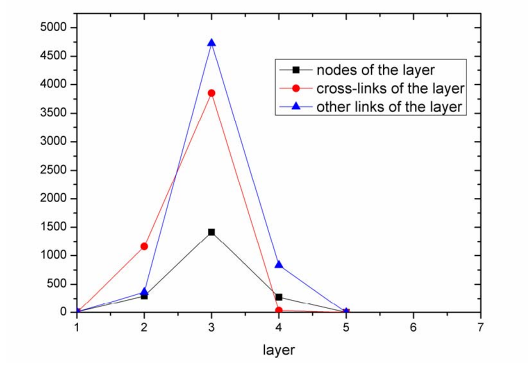


Figure 4. Correlation between nodes and edges on each layer.

given two sources *s1* and *s2* with the subgraphs G_{S1} and G_{S2}, if G_{S1} and G_{S2} are exactly the same, the merged graph of G_{S1} and G_{S2} ($G_{S1} \bigcup G_{S2}$) is the same as G_{S1} or G_{S2}; on the contrary, if G_{S1} is entirely different from G_{S2} ($G_{S1} \bigcap G_{S2} = \varnothing$), the size of merged graph is the sum of G_{S1} and G_{S2}. In this section, we will first examine how to minimize the intersection of two sampled graphs based on the feature of traceroute probes, and then we will validate our method using both the simulated graph and the trace data of skitter[1].

A. Minimize the intersection of two sampled graphs

This subsection is devoted to the following problem: Given a traceroute source *s1* and the subgraph G_{S1} induced by it, how to place the second source *s2* so that the intersection of G_{S1} and G_{S2} can be as small as possible, i.e. *s2* can detect more nodes and edges which are invisible from *s1*. It is known that a graph consists of nodes and edges. Usually the coverage of nodes is proportional to the coverage of edges, so we only take the coverage of edges into account and assume that all nodes in the original graph are destinations. At the end of the section, we will also consider the situation when only parts of the nodes are destinations. From the previous section, we know that the two origins of the undetected edges are cross links and equivalent shortest paths. We will analyze them respectively.

1) Cross-link

Cross-link is the link joining two nodes of the same layer and thus cannot be detected by the source. If *s2* is placed on layer *N*, cross-links on layer *N* can be detected by the exploration from *s2* to the other nodes on layer *N*. Since the number of cross-links on a certain layer is proportional to the number of nodes on the layer, which is demonstrated in fig. 4, we can place *s2* on the layer that contains the maximum number of nodes, i.e. *LMN*(*s1*).

2) Equivalent shortest paths

Edges of a network can be divided into two sets: edge joining two nodes of the same layer and edge joining two nodes belonging to adjacent layers. The former one is actually the set of cross-links and none of these edges can be detected by traceroute probes from a single source. Edges in the latter set can partly be detected. If more than one node in the same layer is joined to a node in the next layer, there will be

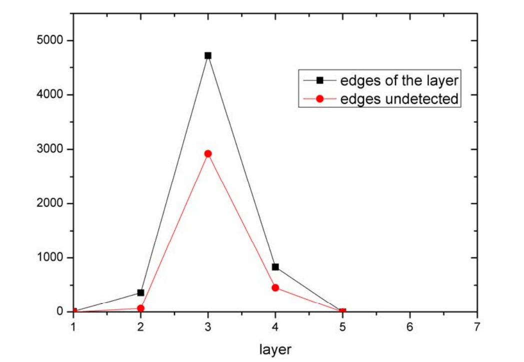


Figure 5. "Edges of the layer" represents the total number of connections to adjacent layer; "edges undetected" represents the number of undetected connections to adjacent layer

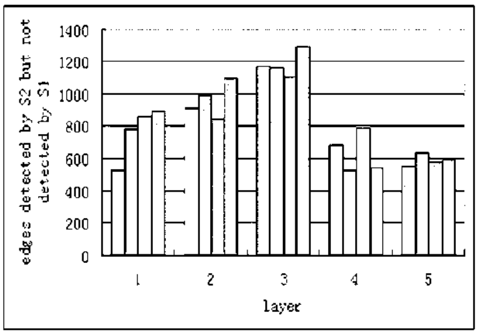


Figure 6. Comparison of number of links detected between s1 and s2

equivalent shortest paths to the node and only one of these paths can be detected by traceroute probes. In fig. 3, node *A* and *B* in layer 1 are both connected to *D* in layer 2, so there are two equivalent shortest paths from *source* to *D*. The statistic of the latter set on each layer in fig. 5 demonstrates that the undetected edges of each layer are proportional to the total edges. Lakhina et al.[7] show that traceroute probes are more likely to find edges very close to the source. So if *s2* is placed on the layer that has maximum number of connections to its adjacent layer, more edges could be detected.

3) How to place s2

From the analysis of the previous subsections, we can conclude that in order to detect more cross-links, *s2* should be placed on layer *LMN*(*s1*), while if *s2* is placed on the layer that has the maximum number of connections to its adjacent layer, it can detect more edges that are invisible from *s1* due to the existence of equivalent shortest paths. Fortunately, the two layers are always the same, which we demonstrate in fig. 4. Therefore, *s2* should be placed on the layer *LMN*(*s1*) so that it can detect more edges that are invisible from *s1*. To validate our conclusion, we randomly select four sources as *s2* from each layer and run traceroute explorations from them. Fig. 6 plots the comparison results between *s1* and *s2*. Note that the source *s1* in fig. 6 is exactly the same one as in fig. 4 and 5. We can observe that sources in layer 3, which contains the maximum number of nodes (shown in fig. 4), can always detect more links that cannot be detected by *s1*.

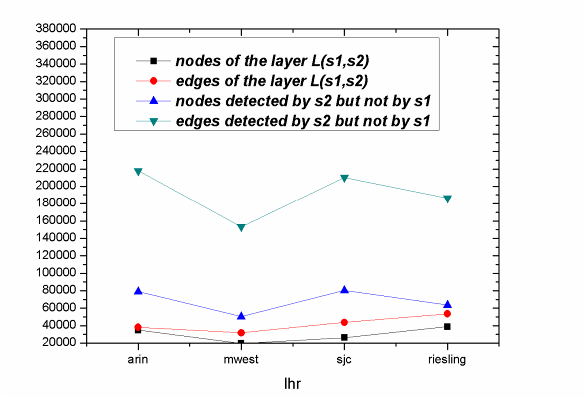


Figure 7. Comparison results between sources. The four curves represent the nodes/links on layer *L*(*s1,s2*) and the nodes/links detected by *s2* but not by *s1*.

Until now, we have made the assumption that all the nodes in the original graph are destinations in the traceroute exploration process. We also validate our conclusion in the case that only parts of the nodes are destinations and the result shows that sources on the layer with the maximum number of nodes can also detect more nodes that *s1* cannot detect.

B. Analysis of raw trace data of skitter

We have reached the conclusion that the second source *s2* should be placed on layer *LMN*(*s1*). In this section, we will validate our conclusion by analyzing the raw trace data of skitter[1].

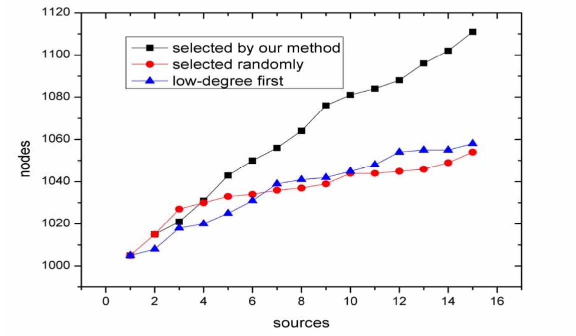
There are 25 sources of skitter running independent probes to more than 971,000 destinations. We choose 5 sources which started a new cycle on Nov 2, 2006 and compare their sampled results. Fig.7 shows the comparison between a certain source(*s1*) and other ones(*s2*). The four curves represents nodes and links detected on layer *L*(*s1*,*s2*), additional nodes and links detected by *s2*. We can see that the four curves exhibit the same trend. Fig. 7 thus supports the conclusion made by us.

VI. THE METHOD TO PLACE SOURCES

A. The method

From the analysis and experiments in section III, we conclude that given a source *s*, another source should be placed on the layer *LMN*(*s*) such that it can detect more nodes and edges that cannot be detected by *s*, and the intersection of their sampled results can be smaller. Based on this conclusion, we propose an iterative method to select sources. The key idea of the method is: whenever adding a new source, the intersections of the sampled results of the new source and each existing source should be as small as possible. The detailed procedure is:

1. Initialization. Randomly select a node from the original graph as the first source *s* and run a traceroute exploration from *s*. Sort all the nodes detected according to their path length from *s*, i.e. their layers. Initialize set *M* to include all the nodes in layer *LMN*(*s*).



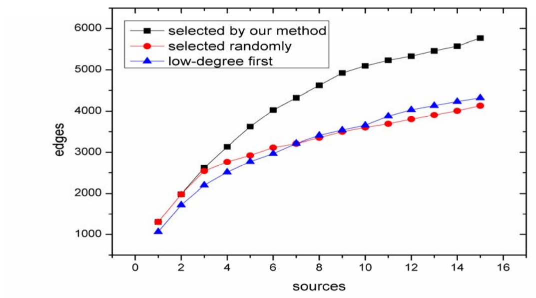


Figure 8. Comparison between three methods of selecting sources, number of desinations=1000

2. If M is not empty, randomly select a node s' from M as the next source, and erase the node from M; else, terminate.

3. Use the source s' selected in step 2 to run a traceroute exploration and sort all nodes detected. Initialize set M' to include all the nodes in layer $LMN(s')$

4. Set M to be the intersection of M and M'. Go to step 2.

B. *Simulation on the modeled graph*

We simulate the traceroute exploration using the sources generated by the method proposed in the previous section and compare it with sources randomly selected. Dall'Asta et al. [12] suggest that traceroute sources should be placed on low-connectivity nodes. So we also simulate the exploration by sources with lowest degrees in the graph. The comparison results are plotted in fig. 8, showing that our method is better than the other two. In case of 15 sources and 1 000 destinations, the sources generated by our methods can detect 57 (2.8% of the original graph) additional nodes and 1 630(14.8% of the original graph) additional links compared with the ones randomly selected.

We also perform the above experiments on graphs generated by other models and topology generators, e.g. the GL model[15], the inet topology generator[19], and even ER random graphs[20]. All the comparison results are similar to the above result.

VII. CONCLUSION

We identify the two reasons that contribute to the low coverage of traceroute-like explorations in this paper. By analyzing the relationship between the placement of traceroute sources and the intersection between the subgraphs induced by the sources, we propose an iterative method on how to place the traceroute sources such that the overall coverage of the exploration can be as higher as possible, i.e. the exploration process can detect as more nodes and edges as possible. We also validate our method on graphs generated by different models.

ACKNOWLEDGMENT

We would like to thank the operators of skitter project for providing the data for analysis. We are also grateful to Yu Gu for his helpful comments and suggestions.

REFERENCES

[1] Cooperative Association for Internet Data Analysis Skitter tool. http://www.caida.org/tools/measurement/skitter/

[2] The National Laboratory for Applied Network Research (NLANR). http://moat.nlanr.net/

[3] Topology project. http://topology.eecs.umich.edu/

[4] SCAN project. http://www.isi.edu/div7/scan/

[5] Internet mapping project at Lucent Bell Labs. http://www.cs.bell-labs.com/who/ches/map/

[6] JL Guillaume, M Latapy, " Relevance of massively distributed explorations of the internet topology: Simulation results", IEEE INFOCOM, 2005.

[7] A Lakhina, JW Byers, M Crovella, and P Xie, "Sampling biases in IP topology measurements", IEEE INFOCOM, 2003.

[8] Shavitt, E Shir, "DIMES: Let the Internet Measure Itself", ACM SIGCOMM Computer Communication Review, 2005

[9] DIMES project. http://www.netdimes.org/

[10] B Donnet, P Raoult, T Friedman, M Crovella, "Efficient algorithms for large-scale topology discovery", ACM SIGMETRICS Performance Evaluation Review, 2005.

[11] P Barford, A Bestavros, J Byers, M Crovella, "On the Marginal Utility of Network Topology Measurements", Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, 2001.

[12] L. Dall'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani, "Exploring networks with traceroute-like probes: Theory and imulations", Theoretical Computer Science, Special Issue on Complex Networks, 2005.

[13] R´eka Albert and Albert-L´aszl´o Barab´asi, "Topology of Evolving Networks: Local Events and Universality", Physical Review Letters, 2000

[14] C. Faloutsos, P. Faloutsos, and M. Faloutsos, "On Power-Law Relationships of the Internet Topology", In Proceedings of the ACM SIGCOMM, 1999.

[15] JL Guillaume, M Latapy, "Bipartite graphs as models of complex networks", Physica A.

[16] M. Molloy and B. Reed, "The size of the giant component of a random graph with a given degree sequence", Combinatorics, Probability and Computing, 1998.

[17] EW Zegura, KL Calvert, MJ Donahoo, "A quantitative comparison of graph-based models for Internet topology", IEEE/ACM Transactions on Networking (TON), 1997

[18] R Siamwalla, R Sharma, S Keshav, "Discovering Internet Topology" IEEE INFOCOM, 1999.

[19] Inet, an Autonomous System (AS) level Internet topology generator. http://topology.eecs.umich.edu/inet.

[20] P Erdös, A Rényi, "On random graphs", I. Publ. Math. Debrecen, 1959.