Probabilistic Diagnosis of Link Loss Using End-to-End Path Measurements and Maximum Likelihood Estimation

Bo Sun and Zhenghao Zhang
Computer Science Department
Florida State University, Tallahassee, FL 32306, USA

***Abstract*—Internet fault diagnosis has attracted much attention in recent years. In this paper, we focus on the problem of finding the Link Pass Ratios (LPRs) when the Path Pass Ratios (PPRs) of a set of paths are given. Usually, given the PPRs of the paths, the LPRs of a significant percentage of the links cannot be uniquely determined because the system is under-constrained. We consider the Maximum Likelihood Estimation of the LPRs of such links. We prove that the problem of finding the Maximum Likelihood Estimation is NP-hard, then propose a simple algorithm based on divide-and-conquer. We first estimate the number of faulty links on a path, then use the global information to assign LPRs to the links. We conduct simulations on networks of various sizes and the results show that our algorithm performs very well in terms of identifying faulty links.**

I. Introduction

Internet fault diagnosis has attracted much attention in recent years. When service was disrupted, the users and the service providers wish to know which router or which link is the cause of the problem. Quite often, the Internet Service Providers (ISPs) are unwilling or unable to collect network measurements [3]. Therefore, the diagnosis has to rely on measurements collected from end-to-end paths, e.g., paths in overlay networks.

In this paper, we focus on the problem of diagnosing Internet link loss with end-to-end path measurements. We consider the problem of finding the Link Pass Ratios (LPRs) when the Path Pass Ratio (PPRs) of a set of paths are given. The common approach to this problem is to establish and solve a set of linear equations [2]. The difficulty is that the set of linear equations is often under-constrained, i.e., the PPR measurements are not always sufficient to uniquely determine the LPR of every link. Therefore, only some links are *identifiable*, i.e., their LPRs can be uniquely determined, while others are *unidentifiable*. To deal with the unidentifiable links, the "Minimal Identifiable Link Sequence" is defined in [3], which is a set of consecutive unidentifiable links that are treated as a single virtual link and are assigned with an aggregate pass ratio. In [5], estimation methods were proposed that "guess" the LPRs of the unidentifiable links.

We note that from a user's point of view, even when the LPR of a link cannot be uniquely determined, it is often desired if some accurate estimations can be provided about the link. In an under-constrained system, although there may exist infinite number of feasible solutions, some solutions are more likely to be true than others. For example, consider the simple network in Fig. 1. Suppose the PPR of paths $A \rightarrow B \rightarrow C$, $A \rightarrow B \rightarrow D$ and $A \rightarrow B \rightarrow E$ are all 0.8. Also suppose that in this network, the LPR of a link is either 1 and 0.8 with probability 0.9 and 0.1, respectively. Given these three measurements, none of links is identifiable, as both configurations lead to the same measurement results. However, clearly, the likelihood of configuration 1 should be much greater than configuration 2. In other words, it is most likely that link AB is the faulty link.

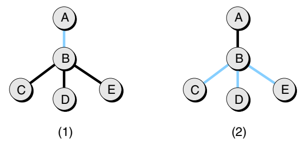


Fig. 1. Two possible configurations that give the same PPR measurements. Lines in black represent links with LPR 1 and lines in light blue represent links with LPR 0.8.

Therefore, in the paper, we study estimation methods. We focus on the Maximum Likelihood Estimations (MLEs) of the LPRs of the unidentifiable links, based on the PPR measurements and the probability distribution function (p.d.f.) of the LPRs in the network. MLE method is chosen because it is the commonly used method in many applications, and, as can be seen in our simulations, it gives very good results even when the solution only approximates MLE. The key assumption in our paper is that we assume the p.d.f. of the LPRs in the network is given, which we believe is reasonable in practice, since the network operators can can obtain the p.d.f. based on the observed history of the network. This is also the key difference between our work and the prior work in [5], which does not make use of this information. We conducted simulations and the results show that our estimation algorithm performs very well in identifying the faulty links.

The rest of the paper is organized as follows. Section II gives a formal definition of the LPR estimation problem. Section III proves that finding MLE is NP-hard. Section IV describes our algorithm for estimating the LPRs. Section V gives the simulation results. Section VI concludes this paper.

II. Problem Formalization

In this section we give a formal definition of the Link Path Ratio Estimation problem. In practice, the PPR is measured by sending probe packets from one end of the path to the other end of the path. If a packets were sent while only b

T	Total number of paths
H	The set of links
P_a	A path
p_a	Path Pass Ratio (PPR) of P_a
L_{ij}	The link between node i to node j
l_{ij}	The Link Pass Ratio (LPR) of link ij
S_a	The set of links on path P_a
F	The distribution of LPRs
e	The estimation of the number of faulty links on a path
Ψ	The set of valid combinations of LPRs

TABLE 1
LIST OF NOTATIONS

packets arrived at the other end, the PPR is defined as b/a. The LPR of a link is defined as the ratio of the probe packets that successfully traveled through the link over all probe packets arrived at this link. As in [2], [3], the LPRs of the links are assumed to be independent of each other. As a result, the PPR of a path is the product of the LPRs of the links on the path.

Given the initial PPR measurements, the LPRs of the identifiable links can be found. For the estimation problem, only the unidentifiable links need to be considered. Therefore, if a path consists of only identifiable links, it is removed from the set of paths. If a path contains some identifiable links, its PPR is adjusted, i.e., divided over the LPRs of these links. Note that after removing the identifiable links, the links on a path may no longer be consecutive, and a path should be considered as a set of links.

We use P to denote a path and L to denote a link. We use p_a to denote the (adjusted) PPR of path P_a and l_{ij} to denote the LPR of link L_{ij}. In addition to the PPR measurements, F is also given which is the p.d.f. of LPRs in the network. With the measurements and F, we wish to estimate the LPRs of the unidentifiable links. In this paper we focus on the Maximum Likelihood Estimation (MLE). Suppose there are T PPR measurements of path P_1, P_2, …, P_T. Denote the set of links on P_a as S_a and denote the set of all links as H. An MLE of the LPRs is a set of values $\{l_{ij}\}$ such that

$$p_a = \prod_{L_{ij} \in S_a} l_{ij} \tag{1}$$

for all $1 \leq a \leq T$ while

$$OBJ = \prod_{L_{ij} \in H} F(l_{ij}) \tag{2}$$

is maximized. Note that $F(l_{ij})$ is the probability that the LPR of L_{ij} takes value l_{ij}. $\prod_{L_{ij} \in H} F(l_{ij})$ is thus the likelihood of the LPRs of the links to take these values. Finally, we define the Link Path Ratio Estimation (LPRE) problem as finding the MLE of the LPRs. The notations that are used in this paper are listed in Table 1.

III. NP-HARDNESS

In this section we prove that the LPRE problem is NP-hard.

Theorem 1: The Link Path Ratio Estimation (LPRE) Problem is NP-hard.

Proof: . We will use the Maximum Independent Set problem. In a graph, a set of vertices that are not connected by an edge is called an *independent set*. The independent set of maximum cardinality is called the *maximum independent set*. Finding the maximum independent set in a graph is known to be NP-hard.

Given any graph G with n vertices, we construct an instance of the LPRE problem as follows. First, for every vertex in G, a new link is created, as shown in Fig. 2. The newly constructed graph is referred to as G'. Vertices in G are referred to as the original vertices, and links in G are referred to as the original links. In G', consider two types of paths: (1) the path consisting of only a single link where the link is an original link, such as path $a \rightarrow b$, and (2) the path starting from a newly added link, traversing an original link, and terminating at a newly added link, such as path $a' \rightarrow a \rightarrow b \rightarrow b'$. In G', PPRs are measured for all such paths. Let the PPRs of all the first type of paths be 1. Let the PPRs of all the second type of paths be θ where $\theta < 1$. Let the LPRs in G' follow distribution $F()$, where $F(1) = p$ and $F(\theta) = 1 - p$ for some $p > 1/2$.

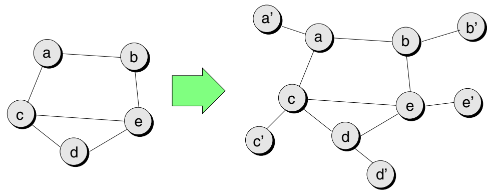


Fig. 2. The construction of the graph.

We claim that any solution to the LPRE problem for the constructed instance defines a maximum independent set in G. This is because given the PPR values and $F()$, among the newly added links in the second type of path, one must have an LPR of 1 and the other must have an LPR of θ. For example, on path $a' \rightarrow a \rightarrow b \rightarrow b'$, either $l_{a'a} = 1$ and $l_{bb'} = \theta$, or $l_{a'a} = \theta$ and $l_{bb'} = 1$. We call the link with an LPR of 1 *selected*. Therefore, for any two adjacent vertices in G, among the two newly added links incident to them, one must be selected and the other must not. If an edge is selected, we say the original vertex it is incident to is selected. Clearly, the set of selected vertices is an independent set. If y links are selected, the objective function defined in Equation 2 is $p^y(1-p)^{n-y}$. Because $p > 1/2$, a larger y gives a larger likelihood. Hence, the solution to the LPRE problem defines a maximum independent set in G. Similarly, it can be easily verified that any maximum independent set in G defines a solution to the LPRE problem. ■

IV. THE LPR ESTIMATION ALGORITHM

Since finding the MLE is NP-hard, in this section, we focus on a simple algorithm that gives reasonably good estimations. Our algorithm is based on two assumptions. First, as mentioned earlier, we assume packet loss is independent among the links. Second, we assume that the majority of the links are good links that do not lose packets, as measurements in [3] found that over 70% of the links are good links.

A. Removing Identifiable Links

As mentioned earlier, before estimating the LPRs of the unidentifiable links, the LPRs of the identifiable links should be determined and such links should be removed. We use a simple method to find the identifiable links. Basically, we consider the *easily identifiable paths* which consists of two types of paths: (1) the paths whose PPRs are 1 and (2) the paths consisting of only one link. Clearly, all links on the first type of paths are identifiable. A link on the second type of path is also identifiable with its LPR equal to the PPR of the path. We remove all links on all such paths and adjust the PPRs of the remaining paths. If there are some new paths become easily identifiable, we again remove links on such paths, until no paths are easily identifiable.

Note that after this, there might still be some identifiable links. In our algorithm, we do not pursue further to find such links, which is primarily because the number of such links is usually very small, i.e., less than 1%, in our simulations. Removing them typically does not result in noticeable performance improvements. Further more, finding such links requires operations on large matrices, which are computationally intensive.

B. Estimating LPR

Recall that because the losses on links are assumed to be independent, the PPR of a path P_a is $p_a = \prod^{L_{ij} \in S_a} l_{ij}$, where S_a denotes the set of links in P_a. After removing the identifiable links, if there are T paths left, there will be T such equations.

We propose a heuristic to estimate of the LPRs. Basically, we pick a path equation, then solve this equation in two steps. In Step 1, we find the *number* of the faulty links on this path using only this path equation. In Step 2, we assign LPRs to links on this path based on the global information in all T path equations. Note that the MLE of the LPRs should give best results, but is computationally infeasible because it jointly considers all T PPR measurements. The other extreme is to estimate LPRs of the links on each path separately without using any global information. Such estimation, of course, is inaccurate and is even useless, because it could lead to inconsistencies across estimations on different paths. Our approach basically represents a compromise between these two extreme cases. We explain our algorithm in details in the following.

1) Step 1: We first estimate the number of faulty links on a path, say, P_a. Let e_a represent the estimation of the number of faulty links on P_a. To get e_a, let $\lambda_{a,t}$ be the likelihood that there are t faulty links on P_a, where $1 \leq t \leq |S_a|$. We set $e_a = t$ if $\lambda_{a,t}$ is maximum among all possible values of t.

To obtain $\lambda_{a,t}$, we use a brutal-force search. Note that there should be exactly t LPR values among the links in S_a that are not one. Let $\Psi_{a,t}$ denote the set of *valid* combinations of t LPR values, where a combination of LPR values $C = \{l_1, l_2, \ldots, l_t\}$ is valid if $\prod_{i=1}^{t} l_i = p_a$. Basically, to calculate $\lambda_{a,t}$, we iterate through all combinations in $\Psi_{a,t}$. For a combination $C = \{l_1, l_2, \ldots, l_t\}$, we calculate the likelihood of C as $[\prod_{i=1}^{t} F(l_i)] F(1)^{|S_a|-t}$. $\lambda_{a,t}$ is set to be the maximum likelihood among all combinations in $\Psi_{a,t}$.

In practice, it is computationally impossible to run the brutal force search "as it is," because the number of combinations $\Psi_{a,t}$ can be infinite. To cope with this, our heuristic takes several simplifications. First, we only consider LPR values at discrete points. That is, our heuristic assumes that the LPRs can take values only among $\{0, 0.01, 0.02, \ldots, 0.99, 1\}$. Therefore, for a small value of t, the search space is actually very small. It could happen that because of the discretization, for some value of p_a, no LPR values can result in exactly p_a. We therefore relax this requirement and considers a set of LPR values valid if the resulting PPR differs with p_a by no more than 0.001. Such discretization could lead to inaccuracies in estimating the LPR, but usually lead to accurate identifications of the faulty links. Second, we actually only run the search for t up to c, where c is a constant set to be 4 in our algorithm. It is possible that there are more than c faulty links on a path, and we will explain our reasons for this treatment shortly. Note that we actually have the freedom to choose which equation to solve first. In out algorithm, the equation with the maximum PPR is solved first.

2) Step 2: After determining e_a, our second step is to assign LPRs to the links on P_a. Let C be the combination of e_a LPR values that results in the maximum likelihood in Step 1. To assign the LPR values, we use the global information. We count the number of appearances of the links in S_a in the T equations, and the link with the i_{th} most appearances will be assigned with the i_{th} smallest LPR values in C for $1 \leq i \leq e_a$. The LPRs of the rest of the links are set to be 1. After determining the LPRs, this equation is *solved*, and all links in S_a will be removed from all paths, and the PPRs of the remaining paths will be adjusted. This process is repeated until all equations are solved.

3) Discussions: In Step 1, we solve the equation with the maximum PPR first. Note that if the PPR is large, it is likely that this path contains only a small number of faulty links. Recall that in Step 1, we only search t up to $c = 4$. Therefore, our search is very likely to be valid for such paths. Then, after this equation is solved, the PPRs of the remaining paths are adjusted. It is likely that after the adjustment, the PPRs of some paths are increased. Therefore the algorithm is actually gradually increasing the PPRs of the equations, and setting $c = 4$ most likely will not hurt the performance.

In Step 2, we assign smaller LPR values to links with more appearances. The reason for this is that we assume the majority of the links are good links, hence, assigning smaller LPR values to links with more appearances will likely reduce the total number of faulty links, and hence result in LPR assignments with a larger likelihood.

We summarize our algorithm as follows.

Algorithm 1 LPR Estimation Algorithm

```
1: while not all equations are solved do
2:     Let $P_a$ be the path with the largest PPR.
3:     Estimate $e_a$ using the method described in Section IV-B1.
4:     Assign LPRs to links in $S_a$ using the method described in Section IV-B2.
5:     Remove all links in $S_a$, adjust the PPRs of the unsolved paths.
6: end while
```

V. Evaluations

In this section, we evaluate our algorithm by simulations. We first ran simulation on randomly generated networks represented by graphs. A graph is generated in two steps. First, we randomly generate edges in the graph. To control the node degree, if the degree of a node is 4, no edge will be generated incident to this node. This step is repeated until no more edges can be generated. Second, we check if every node can reach every other node within 20 hops to make sure that path length in the graph is not excessively long. Basically, we find A^{20} where A is the binary adjacency matrix of the graph. If some elements in A^{20}, say, $A^{20}[i,j]$, is zero, node i cannot reach node j within 20 hops. In this case, we set $A[i,j]$ to be 1, i.e., add an edge between node i and node j. Typically, very few edges need to be added. Each data point was obtained after running the algorithm on 100 independently generated networks.

In order to evaluate our algorithm, a function must be chosen as the p.d.f. of the LPRs. We assume that there are three classes of links in the network. Class 0 links are good links that do not lose packets. Class 1 links occasionally lose packets, but have a reasonable pass ratio. Class 2 are the rest of the links, which could have a very small pass ratio. We assume Class 0 links are of α fraction of the links. We also assume that Class 1 links have LPRs greater than 0.8. According to these assumptions, the function we use as the p.d.f. of the LPR is

$$F(x) = \begin{cases} \alpha & x = 1 \\ ax^2 + bx + c & 0.8 \le x < 1 \\ kx + m & 0 \le x < 0.8 \end{cases},$$

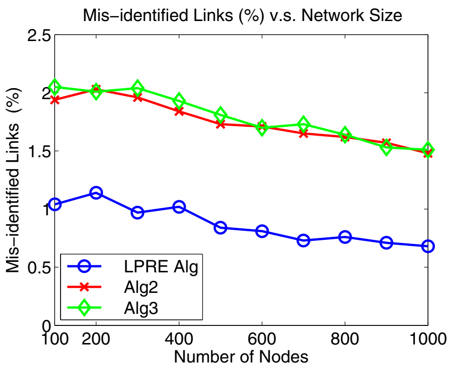
where (α, a, b, c, k, m) are constants. For example, the values we used for Fig. 3(a) and Fig. 3(a) are

$$(\alpha, a, b, c, k, m) = (0.7, -1.5, -4, -1.4786, 3.3, 0.01).$$

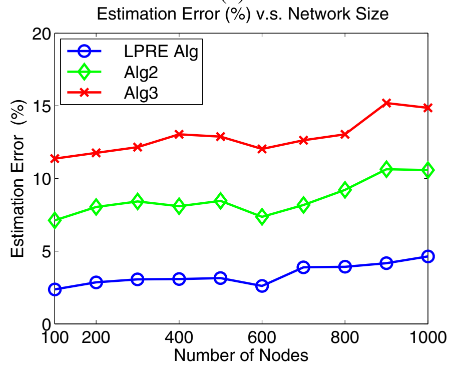
These values were chosen such that (1) there are about 70% of the links that are good links and (2) most of the faulty links have a relatively small loss ratio, matching the measurements in [3]. It should be mentioned that our algorithm should also work with other p.d.f. of LPRs, and is not tied to the specific function we used in our evaluation.

We show the results of our algorithm on random networks of size 100 nodes to 1000 nodes in Fig. 3, where our algorithm is denoted as “LPRE Alg.” Fig. 3(a) shows the percentage of mis-identified links among all links in the network, where a link is mis-identified if the link is a good link but was considered as a faulty link by the algorithm, or if the link is a faulty link but was considered as a good link by the algorithm. Fig. 3(b) shows the average estimation error among all links in the network, where the estimation error of an LPR estimation for link L_{ij} is defined as $\frac{|l_{ij} - \hat{l}_{ij}|}{l_{ij}} \times 100\%$, where $\hat{l}_{ij}$ denotes the estimation of the LPR and l_{ij} denotes the true value of the LPR. From the figures we can see that our algorithm achieves a good performance. For example, the percentage of mis-identified links is less than 2% for all network sizes. Also, the average estimation error is less than 5%.

As a comparison, in addition to our algorithm, we also show the results of two other algorithms referred to as Alg2 and Alg3, respectively. Alg2 differs with our algorithm in that it randomly picks an equation to solve at Step 1, while our algorithm picks the equation with largest LPR first. Alg3 also randomly picks an equation to solve at Step 1, in addition, it also randomly assigns LPRs to the links at Step 2, while our algorithm assigns smaller LPRs to links with more appearances. We can see that the performances of Alg2 and Alg3 are much worse than our algorithm, which justifies the design choices we made.



(a)

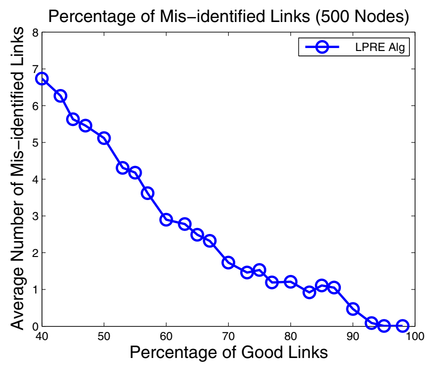


(b)

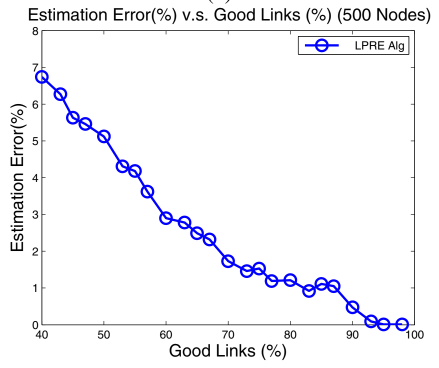
Fig. 3. The performance of the algorithms as a function of the network size. (a) The percentage of mis-identified links. (b) The average estimation error.

It is interesting to notice that the percentage of mis-identified links decreases linearly as the number of nodes grows. This is because in our simulations, a large percentage of links are identifiable and are removed first. Paths consisting of only identifiable links are also removed. The performance of our algorithm is determined by the ratio of the number of the remaining equations oFig. 3(b)ver the number of unidentifiable links. Generally, the larger the ratio, the better the performance. We find that in our simulations, this ratio increases slightly as the number of nodes increases, which leads to the decrease of the percentage of mis-identified links.

Our algorithm is based on the assumption that the majority of links in the network are good links. In Step 2 of our algorithm, we rely on this assumption to assign LPRs to the links. To see the sensitivity of our algorithm to this assumption, we ran simulations on a network with 500 nodes, varying the percentage of good links from 40% to 99%. Fig. 4 shows the percentage of mis-identified links and the average estimation error. We can see that the performance of our algorithm improves faster when the percentage of good links increases

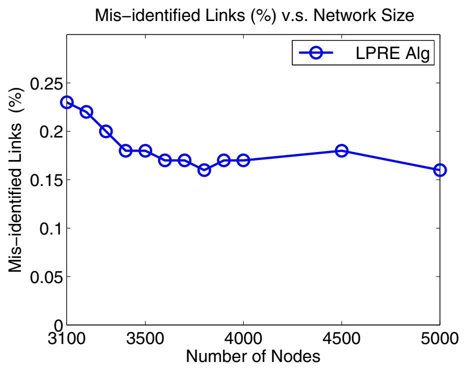


(a)

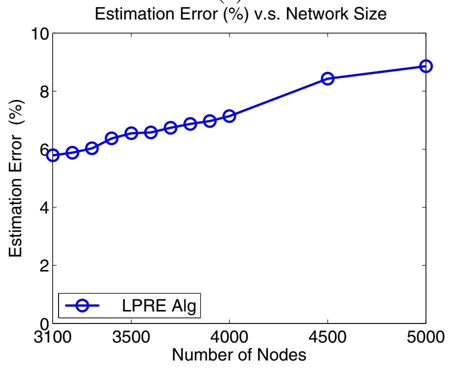


(b)

Fig. 4. The performance of our algorithm as a function of the percentage of good links in a network of size 500. (a) The percentage of mis-identified links. (b) The average estimation error.



(a)



(b)

Fig. 5. The performance of our algorithm as a function of the network size in networks generated by the Inet Topology Generator. (a) The percentage of mis-identified links. (b) The average estimation error.

from 40% to 70%, while the improvement is slower when the percentage of good links is greater than 70%. This suggests that our algorithm works best when the percentage of the good links is over 70%. Interestingly, measurements in [3] found that over 70% of the links are good links in the Internet.

We also carried out evaluations on larger networks generated by the Inet Topology Generator [13], and report our results in Fig. 5, where each data point was obtained by averaging over 20 independent runs. In the simulations, the link pass ratios are generated as in Fig. 3. The numbers of nodes in Fig. 5 are from 3100 to 5000, as the Inet Topology Generator generates network of size no less than 3100 nodes. We can see that in large networks, very few links are misidentified by our algorithm, but the estimation errors tends to increase. These trends are consistent with Fig. 3.

VI. Conclusion

In this paper, we studied the problem of finding the Link Pass Ratios (LPRs) given the measurements of the Path Pass Ratios (PPRs) of a set of paths. We considered the Maximum Likelihood Estimation of the LPRs of unidentifiable links. We proved that the problem of finding the Maximum Likelihood Estimation for the LPRs of such links is NP-hard, then proposed a simple algorithm based on divide-and-conquer. We first estimate the number of faulty links on a path, then use the global information to assign LPRs to the links. We conducted simulations on networks of various sizes and the results show that our algorithm performs very well in terms of identifying faulty links.

References

[1] Y. Zhao and Y. Chen, “ A suite of schemes for user-level network diagnosis without infrastructure,” *IEEE INFOCOM 2007*.
[2] Y. Chen, D. Bindel, H. Song and R. H. Katz, “An algebraic approach to practical and scalable overlay network monitoring,” *ACM SIGCOMM 2004*.
[3] Y. Zhao, Y. Chen and D. Bindel, “Towards Unbiased End-to-End Network Diagnosis,” *ACM SIGCOMM 2006*.
[4] M. Coates, A. Hero, R. Nowak, and B. Yu, “Internet Tomography,” *IEEE Signal Processing Magazine*, vol. 19, no. 3, pp. 4765, 2002.
[5] V. N. Padmanabhan, L. Qiu and H. Wang, “Server-based inference of internet link lossiness,” *IEEE INFOCOM 2003*, San Francisco, CA, USA.
[6] T. Bu, N. Duffield, F. Presti, and D. Towsley, “Network tomography on general topologies,” *ACM SIGMETRICS 2002*.
[7] C. Tang and P. McKinley, “On the cost-quality tradeoff in topology-aware overlay path probing,” *IEEE ICNP 2003*.
[8] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, “Multicast-based inference of network-internal loss characteristics,” *IEEE Transactions in Information Theory*, vol. 45, 1999.
[9] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, 1993.
[10] N. Duffield et al., “Multicast-based loss inference with missing data,” *IEEE Journal of Selected Areas of Communications*, vol. 20, no. 4, 2002.
[11] N. Spring, R. Mahajan, and D. Wetherall, “Measuring ISP topologies with Rocketfuel,” *ACM SIGCOMM 2002*.
[12] Z. Mao, J. Rexford, J. Wang, and R. Katz, “Towards an accurate AS-level traceroute tool,” *ACM SIGCOMM 2003*.
[13] http://topology.eecs.umich.edu/inet/