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

Detecting Anomalies Using End-to-End Path Measurements

K. V. M. Naidu
Bell Labs Research India, Bangalore

Debmalya Panigrahi *
MIT

Rajeev Rastogi
Bell Labs Research India, Bangalore

***Abstract*—In this paper, we propose new "low-overhead" network monitoring techniques to detect violations of path-level QoS guarantees like end-to-end delay, loss, etc. Unlike existing path monitoring schemes, our approach does not calculate QoS parameters for all paths. Instead, it monitors QoS values for only a few paths, and exploits the fact that path anomalies are rare and anomalous states are well separated from normal operation, to rule out path QoS violations in most situations. We propose a heuristic to select a small subset of network paths to monitor while ensuring that no QoS violations are missed. Experiments with an ISP topology from the Rocketfuel data set show that our heuristic can deliver almost a 50% decrease in monitoring overhead compared to previous schemes.**

I. Introduction

Systems for monitoring end-to-end path performance deploy *probes* at strategic locations at the edge of the network (e.g., edge router interfaces connected to customers in a service provider network). These probes can either be *passive* taps [1] that non-intrusively snoop on IP packets traversing network links or *active* machines that simulate synthetic service traffic. For each monitored path, the two probes at the endpoints of the path keep track of the transmitted and received packet counts, as well as the average timestamp values for path packets. At regular time intervals (e.g., every minute), the probes send the count and average timestamp statistics for each monitored path to a central *Network Operations Center* (NOC). At the NOC, the difference between the count and average timestamp values from the two probe endpoints for each path yield the loss and average delay measurements, respectively, for the path.

Our focus in this paper is to use the above probe infrastructure to detect *path delay anomalies*[1]. Thus, given a set of n paths, we would like to know if for any path, its delay exceeds a certain (path-specific) threshold. Further, observe that our probe-based monitoring incurs communication overhead proportional to the number of monitored paths. This is because for each monitored path, the probes need to transmit timestamp information (in each time interval) to the NOC[2]. Thus, to keep communication costs low, we seek to monitor as few paths as possible.

Clearly, an obvious solution to the anomaly detection problem is to simply calculate the latencies of all n paths, and check which of these are above their corresponding thresholds. However, this simple approach requires each and every path to be individually monitored - the communication costs associated with this can be substantial, especially for large networks with hundreds of paths and small time intervals.

Chen et al. [2] propose an alternate approach with much lower communication overhead. Their scheme monitors only a few paths, and then uses the collected path delay measurements to deduce the latencies of the remaining paths. Suppose that there are m links in the network. Now, it is possible to compute the delay for each network link by measuring the delay for m linearly independent paths and then solving the m path equations. Once the individual link delays are known, they can be used to derive the exact delays for the n network paths, and subsequently identify the paths that violate their thresholds. Thus, Chen et al.'s approach requires delay measurements for only m as opposed to n paths. Consequently, since m will generally be much smaller than n, [2]'s scheme can lead to a significant reduction in monitoring overhead.

The work of [2] represents the current state-of-the-art in path performance monitoring. However, we contend that [2]'s scheme is sub-optimal for path anomaly monitoring, and more efficient techniques that monitor fewer than m paths can be devised for detecting path threshold violations. Our claim is based on the following key observations:

1) The objective of path anomaly detection is to identify the paths whose delay exceeds their threshold. - it is not to calculate the exact path delays. For e.g., service quality monitoring applications are primarily interested in finding the paths with extreme delay values that cause service degradation, and not the paths with normal delays.
2) Path anomalies are typically rare events, and for the most part, the system will operate normally. In such a scenario, [2]'s approach of continuously computing delays for all paths (most of which will be normal) can lead to wastage of precious network resources.
3) In [3], the authors argue that anomalies such as delay spikes in paths are *separable*, that is, normal and anomalous path states are well separated. Further, link delay measurements on Sprint's IP backbone network [4] show that 99% of the packets in the backbone experience less than 1msec of delay going through a single router.

Due to the above points, *it is possible to detect path threshold violations without continuously computing all path latencies*.

* This work was done when the author was at Bell Labs Research India, Bangalore.

[1] Adapting our anomaly detection techniques to handle path loss is straightforward.

[2] With active probes, there is also the additional overhead of generating traffic for each monitored path, which can impose a significant burden on the underlying network.

978-1-4244-2026-1/08/$25.00

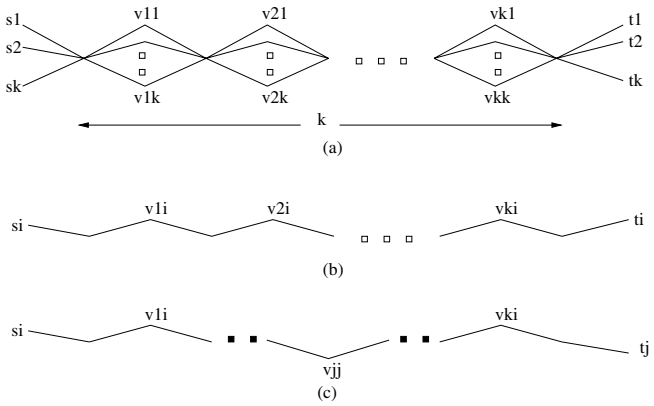

Fig. 1. Example network topology.

In other words, monitoring m linearly independent paths to calculate the delay of each and every path as is done in [2] is sufficient, but not necessary, for detecting path delay constraint violations. As illustrated in the example below, we can eliminate the possibility of path threshold violations (in most cases) without computing individual link and path delays – this enables our approach to monitor much fewer than m paths on an average.

Example 1: Consider the network topology depicted in Figure 1(a). The graph has k source probe endpoints $s_1, \ldots, s_k$, and k destination probe endpoints $t_1, \ldots, t_k$. In addition, there are k stages with the j^{th} stage containing k (2-hop) virtual links passing through vertices $v_{j1}, \ldots, v_{jk}$. Thus, the network graph has a total of k^2 virtual links.

The path set P contains k *basic* paths, one for each source-destination probe pair s_i, t_i (see Figure 1(b)). The i^{th} basic path passes through vertices $v_{1i}, v_{2i}, \ldots, v_{ki}$. In addition to the basic paths, P contains $k-1$ additional *derived* paths for each source s_i. Each derived path for s_i originates at s_i and terminates at a distinct vertex t_j, $j \neq i$. Further, as shown in Figure 1(c), the path connecting s_i and t_j passes through vertices $v_{1i}, \ldots, v_{jj}, \ldots, v_{ki}$. Thus, the path set P contains k^2 paths, each path containing k virtual links.

With high probability, let the delay of each virtual link be less than 1 msec. Thus, whp, each path has a delay of less than k msec. Now suppose that each path has an identical threshold value of $2k$; thus, we are interested in identifying the paths whose delay exceeds twice their normal operating delay.

Observe that since there are k^2 virtual links, the scheme of [2] will end up monitoring all the k^2 paths in P. However, as explained below, we can exploit the gap between normal and abnormal path delays to devise a much more efficient scheme for detecting path delay constraint violations. Our scheme monitors only k paths most of the time, and only in certain rare instances when path delay values deviate significantly from the norm does it monitor k^2 paths.

Our efficient anomaly detection scheme monitors only the k basic paths with a new threshold of k (instead of the original threshold value of $2k$). We consider two cases:

- **Case 1:** If the delay measurements for all k basic paths are $\leq k$, then our scheme simply declares that there are no path anomalies, that is, all paths have a delay below their threshold of $2k$.
- **Case 2:** In the event that the delay for one of the basic monitored paths exceeds k, our scheme monitors k^2 linearly independent paths and derives the individual delays for each and every link and path as in [2]. It then checks if any of the calculated path delays is greater than $2k$, and appropriately flags an anomaly.

Clearly, since normal path delays are less than k msec whp, our efficient scheme monitors only k paths most of the time. For correctness, we need to show that our anomaly detection scheme does not miss any path threshold violations. Here, we need to focus primarily on Case 1 since in Case 2 we compute all the path delays and so there can be no misses. Thus, we need to show that as long as all the basic paths have delay $\leq k$, none of the derived path delays can exceed $2k$. This is easy to see because every derived path is covered by 2 basic paths (the derived path between s_i and t_j is covered by the i^{th} and j^{th} basic paths). Thus, as long as the sum of the delays of all basic path pairs does not exceed $2k$ (which is the case if each basic path has delay at most k), we can be sure that all path delays are less than $2k$. Hence, compared to [2], our efficient scheme can reduce the monitoring overhead by a factor of approximately k, for an arbitrary constant k. □

Our contributions. In this paper, we propose a novel approach for detecting path threshold violations that monitors fewer paths compared to the state-of-the-art path monitoring scheme of [2]. Our approach exploits the fact that anomalies are rare and anomalous states are well separated from normal states, to rule out path anomalies in most cases, while monitoring much fewer than m paths. Only in certain rare occasions when path delays deviate significantly from their normal values does it compute the actual path delays. Our main contributions can be summarized as follows:

- PATH ANOMALY DETECTION FRAMEWORK. We propose a general path monitoring framework for detecting path anomalies without computing the actual delays for all paths.
- PATH SELECTION HEURISTIC. We state that our path selection problem is *NP-hard* and propose a sample-based heuristic to select a low-cost path set to monitor.
- EXPERIMENTAL EVALUATION. We compare the monitoring costs of the path sets generated by our heuristic and the path monitoring scheme of [2] for an ISP topology from the Rocketfuel data set [5]. In our empirical study, we found that our path selection heuristic results in close to 50% fewer paths being monitored compared to the best known scheme for detecting anomalies.

II. RELATED WORK

There is a vast body of research on *network tomography* which involves estimating link-level parameters like delay and loss from end-to-end path-level measurements. However, none

of this work addresses the problem of choosing a small subset of paths to monitor for the purpose of tracking anomalous paths. Instead, their focus is on using statistical algorithms or heuristic methods for either computing delays for smaller path segments [6], or inferring link-level loss rates [7], [8], or identifying links with high loss rates [3], [9], [10].

[11] adopts a very similar approach to [2] to detect anomalies – for a given set of paths, [11] proposes a polynomial-time algorithm for determining the smallest independent subset of paths that can distinguish the same anomalous-link sets as distinguished by the (bigger) input path set. And finally, in recent work, Agrawal et al. [12] advocate monitoring a path cover, that is, a small set of paths that covers all the network links, to detect link anomalies. However, simply monitoring a path cover may be inadequate for detecting path anomalies – this is because setting path threshold values to be large can lead to certain anomalies going undetected (false negatives) while small path thresholds can result in increased communication overhead due to a large number of spurious threshold violations (false positives). Thus, as we shall see later, to efficiently detect path anomalies, it may be necessary to monitor additional paths over and above the minimal path cover.

III. Problem Formulation

A. Network Model

We model the network as a directed graph $G = (V, E)$ where V is the set of nodes (e.g., routers, probes), and $E = \{e_1, \ldots, e_m\}$ is the set of directed links connecting the nodes in V. Let $P = \{p_1, \ldots, p_n\}$ is the set of paths that are to be monitored. Let d_j denote the delay for link e_j; thus, the end-to-end delay for path p_i is the sum of its individual link delays, that is, $\sum_{e_j \in p_i} d_j$. Each path p_i has an associated threshold τ_i which is the delay threshold beyond which service quality would be severely degraded. We are interested in detecting all instances when the latency of a path $p_i \in P$ exceeds its delay threshold τ_i.

The delay constraint for each path p_i has the form $\sum_{e_j \in p_i} d_j \leq \tau_i$. Thus, intuitively, the path delay constraints define a closed region R in the m-dimensional space corresponding to the m network link delays (since link delays are non-negative, we only consider the subspace satisfying $d_j \geq 0$ for all e_j). This region R is enclosed by hyperplanes corresponding to the various path constraints. Detecting a path constraint violation or a path anomaly thus involves finding all instances when the m-dimensional point $\mathbf{d} = (d_1, \ldots, d_m)$ corresponding to the delays for the m links lies outside the region R. In the remainder of the paper, when it is clear from context, we will denote the normal delay region R defined by paths p_i and their thresholds τ_i by the set $\{(p_i, \tau_i) : p_i \in P\}$.

B. Path Selection Problem

In our approach, we monitor a set $Q = \{q_1, \ldots, q_k\}$ of $k < m$ paths (from P) with a new delay threshold γ_i for each path q_i. Suppose that M denotes the monitored region enclosed by hyperplanes $\sum_{e_j \in q_i} d_j \leq \gamma_i$. We conclude that there is no anomaly if the delay f_i for every monitored path q_i is $\leq \gamma_i$, that is, the point $\mathbf{d}$ corresponding to link delays is contained in M; else, we monitor m linearly independent paths as in [2] to compute all path delays, and then check if the delay constraints are satisfied for all the paths. To ensure correctness and low monitoring overhead, the monitored path set $Q \subseteq P$ must be chosen carefully.

Correctness. Correctness requires that no path anomalies are missed by our anomaly detection procedure. This imposes the following constraint on the monitored region M:

Claim 2: The above discussed procedure detects all path anomalies if and only if $M \subseteq R$. □

Next, we define the cost of monitoring in our approach as the expected number of paths monitored.

Monitoring cost. Let $\Pr[M]$ be the probability that $\mathbf{d}$ lies in M. Then, the monitoring cost of our procedure is $\Pr[M] \cdot k + (1 - \Pr[M]) \cdot m$, which is essentially the expected number of paths monitored by it.

Our objective is thus to select a minimum cost path set Q and new thresholds γ_i to monitor subject to the correctness criterion $M \subseteq R$.

Path Selection Problem: Given a set of paths P and delay threshold τ_i for each path $p_i \in P$, compute the set $Q \subseteq P$ of monitored paths and threshold γ_i for each path $q_i \in Q$ such that (1) $M \subseteq R$, and (2) the expected communication cost $\Pr[M] \cdot |Q| + (1 - \Pr[M]) \cdot m$ is minimum. □

IV. Sample-Based Path Selection Heuristic

The path selection problem can be shown to be NP-hard by doing a reduction from the set cover problem [13]. In light of the difficulty of path selection, we consider heuristics to tackle the problem. A key challenge when developing heuristics is to estimate the probability of the monitored region M. Our heuristic draws sample points S from the link delay space to estimate the joint probability distribution of link delays and we rely on the following while doing so:

1) In [4], the authors identify three contributing factors to single-hop delay through operational routers in Sprint's IP backbone: transmission delay, minimum router transit time, and queueing delay. The first two components are simple linear functions of packet size and output link speed, while [4], [14] report that the queueing delay distribution is approximated by a Weibull distribution (with shape parameter b close to 0.6). Thus, we can use the Weibull distribution to calculate the probability density function of single-link delay.
2) Prior studies [2], [7] claim that packet loss dependence among links is highly unlikely due to traffic and path diversity in the Internet. A similar argument can also be used to show that link delays are independent, and we can thus compute the joint link delay probability distribution as the product of individual single-link delay distributions.

Our sample-based heuristic uses the samples to formulate an integer program over a finite set of candidate paths to monitor. The objective function seeks to minimize the expected communication cost, while additional constraints are used to enforce the correctness criterion. A key advantage of the sample-based heuristic is that it adopts a global approach to path selection - however, it may have high computational complexity because accurately capturing the probability distribution in high-dimensional delay space could require large sample sizes.

Let S be a set of sample points in the link delay space that captures the joint probability distribution of link delays. Since link delays are independent, each sample point can be generated by drawing link delay values independently based on their individual probability density functions. Let S_{in} (similarly S_{out}) be the sample points in S which are inside (outside) of R. Note that since anomalies are rare, almost all the sample points in S are inside R. Thus, $|S| \approx |S_{in}|$. Now, let S'_{in} be the set of sample points in M. Clearly, if $M \subseteq R$, then $S'_{in} \subseteq S_{in}$ (but not vice-versa). Now, using the sample points, $\hat{C} = (k \cdot |S'_{in}| + m \cdot (|S_{in}| - |S'_{in}|))/|S_{in}|$ gives an estimate of the expected cost $C = k \cdot \Pr[M] + m \cdot (1 - \Pr[M])$.

Our sample-based heuristic has two phases. In the first phase, the objective is to select paths and thresholds using the sample points so as to minimize the expected cost $\hat{C}$ subject to the constraint $S'_{in} \subseteq S_{in}$. However, the solution obtained does not necessarily satisfy our correctness criterion since $S'_{in} \subseteq S_{in}$ does not guarantee that $M \subseteq R$. To ensure that no anomalies are missed, we have a second phase of the heuristic where we use a greedy strategy to add more paths with corresponding thresholds to the set of monitored paths until $M \subseteq R$.

a) Phase I: Consider an arbitrary path p_i. Clearly, the only interesting thresholds γ for p_i are the ones for which some sample point in S lies on the hyperplane $\sum_{e_j \in p_i} d_j = \gamma$ defined by p_i and γ. Now, represent each such *interesting* threshold by the set of sample points in S which do not satisfy the threshold. Clearly, for each path, there are at most $|S|$ interesting thresholds. Thus, there are a total of at most $n \times |S|$ subsets for the n paths. Each of these subsets corresponds to a unique (path, threshold) pair. Let us call these subsets $\mathcal{S} = \{S_1, S_2, \ldots\}$. Now, our problem can be stated as the integer program given in Figure 2.

In this program, x_{S_i} denotes whether a subset $S_i \in \mathcal{S}$ is included in the solution or not. If $x_{S_i} = 1$, then the corresponding (path, threshold) pair is included for monitoring. Thus, $S'_{in} = S - (\cup_{x_{S_i}=1} S_i)$. x_e denotes if a particular sample point $e \in S_{in}$ is part of any selected subset. Thus, if $x_e = 1$, then $x_e \notin S'_{in}$. Now, constraint (1) ensures that for each selected subset S_i, all sample points in it have $x_e = 1$ and are thus excluded from S'_{in}. Constraint (2) ensures that each sample point in S_{out} is included in at least one selected subset. k is a variable which captures the number of subsets which have been selected, or in other words, the number of

minimize $(k \cdot (|S_{in}| - \sum_{e \in S_{in}} x_e) + m \cdot \sum_{e \in S_{in}} x_e)/|S_{in}|$

subject to:

$$x_e \geq x_{S_i}, \ \forall S_i \in \mathcal{S}, e \in S_{in} \cap S_i \quad (1)$$

$$\sum_{S_i \ni e} x_{S_i} \geq 1, \quad \forall e \in S_{out} \quad (2)$$

$$\sum_{S_i \in \mathcal{S}} x_{S_i} = k \quad (3)$$

$$x_{S_i} \in \{0,1\}, \ \forall S_i \in \mathcal{S} \quad (4)$$

$$x_e \in \{0,1\}, \ \forall e \in S_{in} \quad (5)$$

Fig. 2. Integer Program for path selection problem.

Procedure SAMPLINGPHASE1 (R, S)
Input: Normal region $R = \{(p_i, \tau_i) : p_i \in P\}$, and sample S.
Output: Set of paths to monitor and corresponding thresholds.
begin
1. $M = \emptyset$;
2. $cost = m$;
3. **for** $k = 1$ to m {
4. Solve LP to obtain optimal fractional x_e, x_{S_i} values;
5. **while** $\exists e \in S_{out}$ s.t. $x_{S_i} < 1 \ \ \forall S_i \ni e$
6. Round fractional x_{S_i} values using randomized rounding;
7. **if** $x_{S_i} = 1$ **and** $e \in S_i$ **then** $x_e = 1$ **else** $x_e = 0$;
8. $\hat{C} = (k \cdot (|S_{in}| - \sum_{e \in S_{in}} x_e) + m \cdot \sum_{e \in S_{in}} x_e)/|S_{in}|$;
9. **if** $\hat{C} < cost$ **then** {
10. $M =$ (path, threshold) pairs for sets S_i s.t. $x_{S_i} = 1$;
11. $cost = \hat{C}$;
12. }
13. }
14. **return** M;
end

Fig. 3. Phase I of the sampling-based heuristic.

paths which are to be monitored[3]. With this interpretation, it is easy to see that the objective function is exactly equal to $\hat{C} = (k \cdot |S'_{in}| + m \cdot (|S_{in}| - |S'_{in}|))/|S_{in}|$. To make the objective function linear, we iterate over different values of k ranging from 1 to m fixing k to a constant value in any particular iteration. This gives an integer linear program for this problem. We also relax the integrality constraints to get a linear program.

Figure 3 depicts the procedure for phase 1 of our heuristic. We use an LP solver to obtain a solution to our LP for all values of k ranging from 1 to m, round all the solutions we obtain to get corresponding integral solutions and compare the objective values for all these solutions. The solution having the smallest value of the objective function gives the final set of paths and thresholds that we monitor. In our rounding procedure, for each fractional variable x between 0 and 1, the

[3]Note that any two subsets corresponding to the same path share a subset-superset relationship. Hence, in the optimal solution, only one of these two subsets will be selected. This ensures that the number of selected subsets, k is equal to the number of monitored paths.

following rounding rule is used:

$$round(x) = \begin{cases} 1 & \text{with probability } x \\ 0 & \text{with probability } 1-x \end{cases}$$

Using this rule, we round each variable x_{S_i} corresponding to the subsets $S_i \in \mathcal{S}$. Now, we repeat the rounding procedure independently multiple times and set $x_{S_i} = 1$ if and only if x_{S_i} is rounded to 1 for at least one rounding iteration. We continue repeating the rounding procedure until constraint (2) is satisfied for each sample point $e \in S_{out}$.

b) Phase II: Recall that the solution to the IP in Figure 2 ensures that $S'_{in} \subseteq S_{in}$ (i.e., the set of sample points satisfying all the new thresholds also satisfy the original path delay constraints). However, as pointed out earlier, this does not guarantee our correctness criterion, which is $M \subseteq R$. This necessitates a second phase of the sampling-based heuristic. In this phase, we greedily and iteratively add new paths and thresholds to the monitored set until $M \subseteq R$ such that S'_{in}, which is the set of sample points in M, remains unchanged. This ensures that the increase in the expected cost is only due to an increase in the number of monitored paths, and so our goal in the second phase is to select as few additional paths as possible.

Let us define the *interesting* threshold δ_i for each path p_i not included in Q as the maximum of the delays corresponding to the sample points in S'_{in} for p_i. In each iteration of our greedy procedure GREEDYPATH, a path p_i and its interesting threshold δ_i are added to the monitored region M greedily as follows. For each path $p_i \notin Q$, we first add (p_i, δ_i) to M to form M_i. Next, we find the number of original path delay constraints that are violated by points in M_i. Intuitively, the number of original path delay constraint violations is a good indicator of how close M_i is to being fully contained in R. Thus, we select the set M_i with the minimum value for this quantity in each iteration of procedure GREEDYPATH. The procedure terminates once $M \subseteq R$ or the number of monitored paths in Q reaches m.

V. EXPERIMENTS

We use the Telstra ISP topology and corresponding average link delay values from the Rocketfuel project data [5] for our experiments. Paths are generated between randomly selected pairs of terminal nodes. In [4], it was observed that link delays follow the Weibull distribution [15] with shape parameter 0.6. We use this observation and adjust the scale parameter of the Weibull distribution for each link so that the expected link delay given by the Weibull distribution matches the average link delay in the Rocketfuel data. Now, let us define the *link threshold violation probability* corresponding to a delay threshold T for a link as the probability (given by the fitted Weibull distribution for the link) of the link delay exceeding T. Since a threshold violation or an anomaly is a rare event, we run our experiments with low link threshold violation probabilities ranging from 10^{-7} to 10^{-3}. We run our experiments for two different scenarios, once with 104 links and 150 paths and in the other case, with 118 links and 300 paths. The expected number of monitored paths is compared to that of the previous best algorithm [2]. The results are shown in Figure 4. We achieve a performance improvement of nearly 50% compared to current state-of-the-art techniques [2].

Link threshold violation	Improvement (in %) over m paths	
	104 links, 150 paths	118 links, 300 paths
10^{-7}	47.68	45.39
10^{-6}	47.68	43.44
10^{-5}	41.06	34.05
10^{-4}	43.56	27.03
10^{-3}	38.81	29.54

Fig. 4. Improvement (in %) over monitoring m paths by using sample-based heuristic for different link failure probabilities.

VI. CONCLUSION

In this paper, we proposed new techniques for detecting path QoS anomalies without calculating QoS parameters for all paths. Our techniques monitor QoS values for only a few paths, and exploit the fact that path anomalies are rare and anomalous states are well separated from normal operation, to rule out path QoS violations in most situations. We proposed a sampling-based heuristic to compute a small set of paths to monitor; on a real-life ISP topology, this approach reduced the monitoring overhead by nearly 50% compared to existing path monitoring schemes.

REFERENCES

[1] "Net optics: Network taps and aggregation solutions for passive network access." [Online]. Available: http://www.netoptics.com

[2] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scaleable overlay network monitoring," in *ACM SIGCOMM*, 2004.

[3] N. G. Duffield, "Simple network performance tomography," in *IMC*, 2003.

[4] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, F. Tobagi, and C. Diot, "Analysis of measured single-hop delay from an operational backbone network," in *IEEE INFOCOM*, 2002.

[5] [Online]. Available: http://www.cs.washington.edu/research/networking/rocketfuel

[6] Y. Shavitt, X. Sun, A. Wool, and B. Yener, "Computing the unmeasured: An algebraic approach to internet mapping," in *IEEE INFOCOM*, 2001.

[7] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network internal loss characteristics," *IEEE Transactions on Information Theory*, vol. 45, 1999.

[8] T. Bu, N. Duffield, F. L. Presti, and D. Towsley, "Network tomography on general topologies," in *ACM SIGMETRICS*, 2002.

[9] V. N. Padmanabhan, L. Qiu, and H. J. Wang, "Server-based inference of internet performance," in *IEEE INFOCOM*, 2003.

[10] H. Nguyen and P. Thiran, "Using end-to-end data to infer lossy links in sensor networks," in *IEEE INFOCOM*, 2006.

[11] ——, "Active measurement for multiple link failures: Diagnosis in IP networks," in *PAM*, 2004.

[12] S. Agrawal, K. V. M. Naidu, and R. Rastogi, "Diagnosing link-level anomalies using passive probes," in *IEEE INFOCOM*, 2007.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[14] I. Norros, "On the use of fractional brownian motion in the theory of connectionless networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, 1995.

[15] [Online]. Available: http://en.wikipedia.org/wiki/Weibull_distribution