Network Tomography Using Passive End-to-end Measurements

Venkata N. Padmanabhan (Microsoft Research) and Lili Qiu (Microsoft Research)

I. Introduction

Network tomography refers to the inference of characteristics of internal links in a network using end-to-end measurements. The link characteristics of interest include packet loss rate, delay, and bandwidth; here we focus on loss rate. Depending only on end-to-end measurements is convenient in the context of the Internet because network operators such as ISPs offer limited visibility into the internal functioning of their networks.

Besides being an interesting problem in its own right, network tomography can help identify bottlenecks and trouble spots (e.g., points of congestion) within the network. This information can help diagnose network problems and, in the long run, drive network provisioning decisions for ISPs and network connectivity and server placement decisions for their customers such as Web site operators.

Previous work on inferring link loss rate using end-to-end measurements has largely been based on active probing techniques. MINC [1] bases its inference on losses experienced by multicast probe packets injected into the network while [2] does so using closely-spaced unicast probe packets striped across multiple destinations.

In contrast, our goal is to infer link loss rates based on *passively* observing the end-to-end loss rate for existing traffic such as that between a Web server and its clients. A passive approach has the advantage that there is no wasteful traffic and the measurements do not perturb the network. However, the disadvantage is that we have less control over the measurement process. Unlike active techniques that are able to identify and localize individual loss events, our passive approach has to make do with aggregate statistics such as the loss *rate*. While accuracy may suffer, we believe a passive approach is still advantageous if we can infer where the trouble spots (e.g., highly lossy links) are in the network.

While being basically passive, our approach has a small active component to discover the network topology using *traceroute* measurements. However, these measurements only need to be made relatively infrequently and can be done in the background.

II. Identifying Lossy Links

We now discuss the problem of identifying lossy links based on passive measurements. The scenario we focus on is a large number of clients downloading files from a server. The packet loss rate between the server and each client is computed by passively observing the traffic between the server and the clients. The network topology from the viewpoint of the server is constructed by tracing the path to each client. Barring transient route fluctuations, the resulting topology is a tree rooted at the server, with clients at the leaves (Figure 1). Our goal is to identify the lossy links in this topology.

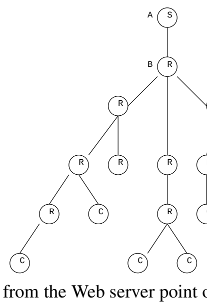

Fig. 1. Network topology from the Web server point of view, where S denotes a server, C denotes a client, and R denotes a router.

Identifying lossy links is challenging for the following reasons. First, network characteristics change over time. Without knowing the temporal variation of the network link performance, it is hard to correlate different clients' performance. Second, even when the loss rate of each link is constant, there may not be a unique explanation for the observed end-to-end loss rates. Given M clients and N links, we have M constraints in N variables (i.e., loss rates of the individual links). For each client C, there is a constraint of the form $1-\prod_{i\in P}(1-l_i) = l_C$ where P is the set of links on the path from the server to the client, l_i is the loss rate of link i, and l_C is the end-to-end loss rate between the server and client C. (We can turn these into linear constraints by taking the logarithm of both sides and defining the variable corresponding to link i to be $L_i = log(1/(1-l_i))$.) There is not a unique solution to this set of constraints if $M < N$, as is often the case.

We now describe our initial approach to the problem. To make the problem tractable, we make the simplifying assumption that the loss rate of each link is constant. Although this is not a very realistic assumption, it is a reasonable simplification in the sense that some links consistently tend to have high loss rates whereas other links consistently tend to have low loss rates. Zhang et al. [4] define a notion of *operational stationarity* by categorizing loss rates into "no loss","minor loss","tolerable loss" and "unacceptable loss". They find that about two-thirds of the time, the loss process remains operationally stationarity for at least an hour's duration.

Furthermore, since there is not, in general, a unique assignment of loss rates to network links, our goal here is to identify links that are likely to have a high loss rate rather than compute a specific loss rate for each link. Below we describe two approaches: (i) random sampling, and (ii) linear optimization. We discuss these in turn.

A. Random Sampling

The basic idea here is to repeatedly sample the solution space at random and make inferences based on the statistics of the sampled solutions. The solution space is sampled as follows. Figure 1. We first assign a loss rate of zero to each link of the tree (Figure 1). The loss rate of link AB is bounded by the minimum of the observed loss rate (say l_{min}) among clients downstream of the link. We assign a random number between 0 and l_{min} to be the loss rate l_{AB} of the link AB. We define the residual loss rates of a client to be the loss rate that is not accounted for by the links whose loss rates have already been assigned. We update the residual loss rate of a client C to $1 - \frac{1-l_C}{\prod_{i\in P'}(1-l_i)}$ where P' is the subset of links along the path from the server to the client C for which a loss rate has been assigned. Then we iterate the procedure to compute the loss rate at the next level of the tree by considering the residual loss rate of each client in place of its original loss rate.

There are a number of details to the algorithm. First, if there are no branches along (a section of) a path, it is impossible to estimate the loss rates of the individual links on (that section of) the path solely using end-to-end measurements. Therefore, we coalesce such a path into a single "link", before running the algorithm. Secondly, we need a large sample of packets to get an accurate estimate of the loss rate at clients. So we filter out the leaves (clients) to which sender sends fewer than a threshold number of packets during the measurement period. We set the threshold to be 1000 packets in our analysis.

The random sampling approach has its shortcomings. First, it tends to spread the blame for the observed loss rates. For example, it may assign a significant loss rate both to the transcontinental link from the U.S. to Japan and to multiple ISP links inside Japan whereas concentrating the blame on the transcontinental link may yield a more parsimonious explanation for the observed loss rates. Second, the algorithm

Date	Duration	# packets	# clients
20 Dec 2000	2.12 hours	100.0 million	134,475
24 Jan 2001	1.23 hours	20.38 million	53,811

TABLE I

Summary of the two traces analyzed in this paper.

is vulnerable to estimation errors (say due to statistical variations) in the client loss rates. For instance, the underestimation of loss rate for a single client in Japan could shift the blame away from the transcontinental link to links lower down in the tree (i.e., links within Japan).

B. Linear Optimization

Our goal is to address the shortcomings of random sampling by (i) allowing the loss rate constraints to be violated to tolerate estimation errors, and (ii) seeking a parsimonious explanation for the observed end-to-end loss rates. We formulate this as a linear optimization problem as follows. The basic idea is to introduce a slack variable, S_j, in each loss constraint j, that helps accommodate measurement errors. Our goal is to minimize the summation of loss rates over all links and the slack variables, $\sum_i L_i + \sum_j |S_j|$, where L_i is transformed link loss rate variable introduced above. Intuitively, this problem formulation seeks a solution that satisfies the original loss rate constraints as closely as possible (i.e., with minimal slack) while concentrating the loss rate on a relatively small number of links.

III. Experimental Evaluation

We evaluate both approaches using real packet traces and simulations. The packet traces were gathered at the *microsoft.com* site by placing a packet sniffing box on the spanning of a Cisco Catalyst switch. We only captured (the headers of) TCP packets since we are able to estimate packet loss rate by observing TCP data packets and the corresponding ACKs.

We detect packet losses by looking for packet retransmissions by the sender. The underlying assumption is that (a) the TCP sender only retransmits a packet if the original transmission was lost, and (b) no packets are lost on the network path between the server node and the packet sniffer. The former assumption is reasonable since TCP is conservative about retransmissions. The latter assumption is likely true because the local network is over-engineered so that it is rarely, if ever, a point of congestion. We compute the loss rate for client node as the ratio of the number of retransmitted packets to the total number of packets sent to it. Table I summarizes the two traces we analyze in this paper.

We used the *traceroute* tool to determine the network path from the *microsoft.com* site to each of the clients seen in the traces.

We ran 500 independent iterations of the randomized algorithm, each time constructing a (likely different) feasible solution to the link loss rate estimation problem. We compute the mean loss rate for each link by averaging over the 500 iterations.

For both the random sampling and the linear optimization approaches, we identify the 50 most lossy links. (Note that because of the coalescing procedure described in Section II, a lossy "link" may actually correspond to a sequence of network links.) For each link identified as lossy, we compute the round-trip time (RTT) of the link by subtracting the RTT reported by *traceroute* for the near end of the link from that reported for the far end. While this is not a very accurate calculation, it suffices for our purposes since we are only trying to classify links as having a large RTT or not. We also determine the autonomous system (AS) corresponding to either end of the link by querying the *whois* database. If the two ends are in different ASes, we classify the link as an inter-AS link (e.g., one that crosses the boundary between two ISPs). Otherwise, the link is classified as an intra-AS link.

Anecdotal evidence suggests that network links with a long delay (e.g., transcontinental links) or ones at the peering point between ISPs are often points of congestion. We check to see how often the identification of lossy links by our algorithms is consistent with this intuition. As shown in Figure 2, of the 50 links identified as most lossy, 42–45 cross an inter-AS boundary and/or have round-trip delay > 100 ms. Examples of lossy links identified include a link from AT&T in San Francisco to Indo.net in Indonesia (long delay and inter-AS crossing), one from Sprint to PacBell in California (inter-AS crossing), and the path within Sovam Teleport (a Russian ISP) between Moscow and Tyumen, Siberia (long delay). While these findings are not conclusive, they are consistent with our intuition regarding where in the network packet loss is likely to occur.

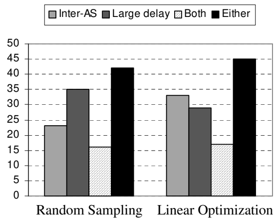

Fig. 2. Characteristics of identified lossy links.

We also evaluate the accuracy of our approaches using simulations. Simulations offer the advantage that there is no uncertainty about the actual link loss rate, so we can compute the estimation error exactly. Our simulations use two types of topologies: randomly generated tree topologies, and the real topology obtained by tracing the Internet paths from the *microsoft.com* site to its clients. We generate 1000-node random tree topologies by varying the maximum node degree from 5 to 50. For each setting of the maximum node degree, we generate 5 different random topologies. We also randomly generate packet loss events in accordance with a Bernoulli process where the loss rate is set to 0–1% for 95% of the links and 5–10% for the remaining 5% of links. Our performance metric is the percentage of links that are classified correctly as either low loss rate ($< 5\%$), or high loss rate ($\geq 5\%$). Our simulation results show that the percentage of correct classification is 90 - 94% for randomly generated topologies, and 85 - 90% for real topologies. Moreover, 70 - 80% lossy links are correctly identified. Along these correctly identified lossy links, there are some non-lossy links that are identified as lossy. This false positive rate is 10–85%. This rate is quite high and we are working to improve our estimation techniques to reduce this false positive rate. However, we believe that the techniques in their current form are still useful, since we can employ these passive techniques to quickly narrow down the set of candidate lossy links and then use more expensive techniques such as active probing to prune out the links that are incorrectly classified as lossy. The relatively small percentage of highly lossy links in the Internet means that expensive active probing only needs to be applied to a small *number* of links even if the false positive *rate* is high.

IV. Conclusion and Future Work

In this paper, we describe two techniques to infer link loss rates based on passive, end-to-end measurements. We evaluate the accuracy of our approaches using real packet traces and simulations. Since then, we have investigated Bayesian inference using Gibbs sampling, and evaluated these techniques using extensive simulations and also Internet packet traces; refer to [3] for further details.

References

[1] R. Caceres, N.G. Duffield, J. Horowitz, and D. Towsley. Multicast-based Inference of Network Internal Loss Characteristics. *IEEE Transactions on Information Theory*, November 1999

[2] N.G. Duffield, F.L. Presti, V. Paxson, and D. Towsley. Inferring Link Loss Using Striped Unicast Probes. *IEEE Infocom*, April 2001

[3] V. N. Padmanabhan, L. Qiu, and H. Wang. Server-based Characterization and Inference of Internet Performance. Microsoft Technical Report, MSR-TR-2002-39, April 2002.

[4] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker, On the Constancy of Internet Path Properties, Proc. ACM SIGCOMM Internet Measurement Workshop (IMW'2001), San Francisco, California, USA, November 2001.