The Use of End-to-end Multicast Measurements for Characterizing Internal Network Behavior*

A. Adams[1], T. Bu[2], R. Cáceres[3], N. Duffield[3], T.Friedman[2], J. Horowitz[4]
F. Lo Presti[3], S.B. Moon[2], V. Paxson[5], D. Towsley[2]

[1] Pittsburgh Super Computing Center, Pittsburgh, PA 15213
[2] Dept. of Computer Science, U. Massachusetts, Amherst, MA 01003
[3] AT&T Labs–Research, 180 Park Ave., Florham Park, NJ 07932.
[4] Dept. of Math. & Stat., U. Massachusetts, Amherst, MA 01003
[5] ACIRI, Berkeley, CA 94704

February 15, 2000

Abstract

We present a novel methodology for identifying internal network performance characteristics based on end-to-end multicast measurements. The methodology, solidly grounded on statistical estimation theory, can be used to characterize the internal loss and delay behavior of a network. Measurements on the MBone have been used to validate the approach in the case of losses. Extensive simulation experiments provide further validation of the approach, not only for losses, but also for delays. We also describe our strategy for deploying the methodology on the Internet. This includes the continued development of the National Internet Measurement Infrastructure (NIMI) to support RTP-based end-to-end multicast measurements and the development of software tools for analyzing the traces. Once complete, this combined software/hardware infrastructure will provide a service for understanding and forecasting the performance of the Internet.

*This work was sponsored in part by DARPA and the Air Force Research Laboratory under agreement F30602-98-2-0238, by DARPA award #AOG205, and by the National Science Foundation under Cooperative Agreement No. ANI-9720674. The Government has certain rights in this material.

1 Introduction

As the Internet grows in size and diversity, its internal performance becomes ever more difficult to measure. Any one organization has administrative access to only a small fraction of the network's internal nodes, whereas commercial factors often prevent organizations from sharing internal performance data. End-to-end measurements using unicast traffic do not rely on administrative access privileges, but it is difficult to infer link-level performance from them and they require large amounts of traffic to cover multiple paths. There is, consequently, a need for practical and efficient procedures that can take an internal snapshot of a significant portion of the network.

We have developed a measurement technique that addresses these problems. *Multicast Inference of Network Characteristics* (MINC) uses end-to-end multicast measurements to infer link-level loss rates and delay statistics by exploiting the inherent correlation in performance observed by multicast receivers. These measurements do not rely on administrative access to internal nodes since they are done between end hosts. In addition, they scale to large networks because of the bandwidth efficiency of multicast traffic.

Focusing on loss for the moment, the intuition be-

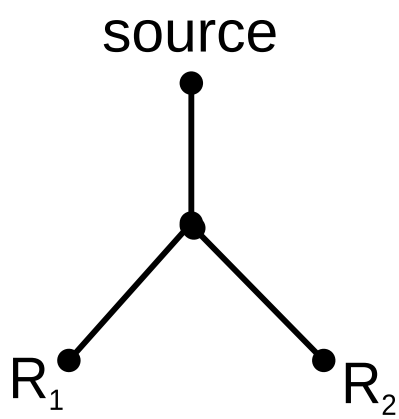

Figure 1: A tree connecting a sender to two receivers.

hind packet loss inference is that the event of the arrival of a packet to a given internal node in the tree can be inferred from the packet's arrival at one or more receivers descended from that node. Conditioning on this latter event, we can determine the probability of successful transmission to and beyond the given node. Consider, for example (Figure 1) a simple multicast tree with a root node (the source), two leaf nodes (receivers R_1 and R_2), a link from the source to a branch point (the shared link), and a link from the branch point to each of the receivers (the left and right links). The source sends a stream of sequenced multicast packets through the tree to the two receivers. If a packet reaches either receiver, we can infer that the packet reached the branch point. Thus the ratio of the number of packets that reach both receivers to the total number that reached only the right receiver gives an estimate of the probability of successful transmission on the left link. The probability of successful transmission on the other links can be found by similar reasoning.

This technique extends to general trees (see [1]) and it can be shown that the resulting loss rate estimates converge to the true loss rates as the number of probes grows indefinitely large. This and related approaches can be used to estimate path delay distributions, [7], the path delay variances, [4], and the logical multicast topology itself [2]. We have validated the accuracy of the loss rate inference techniques against measurements on the MBone. Further validation of both the loss rate and the delay statistics inference techniques has been made through simulation experiments.

In this paper we describe the MINC methodology (Section 2) and the results of the network measurements and simulation experiments (Section 3). Following this, we describe our efforts to deploy this methodology. These include the further development of the National Internet Measurement Infrastructure (NIMI) [10] to support the required multicast measurements, the extension of the RTP control protocol, RTCP, to include detailed loss reports, and the development of the Multicast Inference Network Tool (MINT) to visualize and manipulate the multicast-based inferred internal network performance.

A survey of related work is found in Section 5, and Section 6 offers some conclusions.

2 Statistical Methodology

MINC works on *logical* multicast trees, i.e. those whose nodes are identified as branch points of the physical multicast tree. A single logical link between nodes of the logical multicast tree may comprise more than one physical link. MINC infers composite properties–such as loss and delay–of the logical links. Henceforth when we speak of trees we will be speaking of logical multicast trees.

2.1 Loss Inference

We model packet loss as independent across different links of the tree, and independent between different probes. With these assumptions, the loss model associates with each link k in the tree, the probability α_k that a packet reaches the terminating node of the link, also denoted by k, given that it reaches the parent node of k. The link loss probability is, then, $(1-\alpha_k)$. When a multicast probe is transmitted from the source, we record the outcome as the set of receivers reached by the probe. The loss inference algorithm employs a probabilistic analysis that expresses the α_k directly as a function of the probabilities of all possible such outcomes. We infer the link probabilities by the estimators $\widehat{\alpha}_k$ obtained by using instead the actual frequencies of the outcomes arising from the dispatch a number of probes. The paper [1] contains a detailed description and analysis of the inference algorithm.

The estimators $\widehat{\alpha}_k$ have several desirable statistical properties. It was shown in [1] that $\widehat{\alpha}_k$ is the Maximum Likelihood Estimator (MLE) of α_k when sufficiently many probes are used. The MLE is defined as the set of link probabilities that maximizes the probability of obtaining the observed outcome frequencies. The MLE property in turn implies two further useful properties for $\widehat{\alpha}$, namely (i) *consistency*: $\widehat{\alpha}_k$ converges to the true value α_k almost surely as the number of probes n grows to infinity, and (ii) *asymptotic normality*: the distribution of the quantity $\sqrt{n}(\widehat{\alpha}_k - \alpha_k)$ converges to a normal distribution as n grows to infinity. The latter property implies that the probability of an error ofa given size in estimating a link probability goes to zero exponentially fast in the number of probes.

The computation of the $\widehat{\alpha}_k$ is performed recursively on the tree; the computational cost is linear in the number of probes and number of nodes in the tree.

2.2 Delay Distribution Inference

A generalization of the loss inference methodology allows one to infer per link delay distributions. More precisely, we infer the distribution of the variable portion of the packet delay, what remains once the link propagation delay and packet transmission time are removed. Packet link delays are modeled as discrete random variables that can take one of a finite number of values, independent between different packets and links. The model is specified by a finite set of probabilities $\alpha_k(t)$ that a packet experiences delay t while traversing the link terminating at node k, with infinite delay interpreted as loss.

When a probe is transmitted from the source, we record the outcome at the each of the receivers that the probe reached, and the time taken to reach each receiver. As with the loss inference, a probabilistic analysis enables us to relate the $\alpha_k(t)$ to the probabilities of the outcomes at the receivers. We infer the link delay probabilities by the estimators $\widehat{\alpha}_k(t)$ obtained by using instead the actual frequencies of the outcomes arising from the dispatch a number of probes. In [7], it was shown that the corresponding estimator $\widehat{\alpha}(\cdot)$ of the link delay distributions is strongly consistent and asymptotically normal.

2.3 Delay Variance Inference

A direct method of delay variance estimation has been proposed in [4]. Consider the binary topology of Figure 1. Let D_0 be the packet delay on the link emanating from the source, and D_i, $i = 1, 2$, the delay on the link terminating at receiver i. The end-to-end delays from the source to leaf node $i = 1, 2$, is expressed as $X_i = D_0 + D_i$. A short calculation shows that, with the assumption that the D_i are independent, $\mathrm{Var}(D_0) = \mathrm{Cov}(X_1, X_2)$. Thus the variance of the delay D_0 can be estimated from the measured end-to-end delays from the source to the leaves. A generalization of this approach can be used to estimate link delay variances in arbitrary trees.

2.4 Topology Inference

In the loss inference methodology described above, the logical multicast tree was assumed to be known in advance. However, extensions of the method enable inference of an unknown multicast topology from end-to-end measurements. We describe briefly three approaches.

Loss-Based Grouping An approach to topology inference was suggested in [13], in the context of grouping multicast receivers that share the same set of network bottlenecks from the source. The loss estimator of Section 2.1 estimates the shared loss to a pair of receivers, i.e., the composite loss rate on the common portion of the paths from the source, irrespective of the underlying topology. Since this loss rate is larger the longer the common path in question, the actual shared loss rate is maximized when the two receivers in question are siblings.

A binary tree can be reconstructed iteratively using this approach. Starting with the set of receiver nodes R, select the pair of nodes j, k in R that maximizes the estimated shared loss, group them together as the composite node, denoted $j \vee k$, that is identified as the parent, and construct the set of remaining nodes $R' = (R \cup \{j \vee k\}) \setminus \{j, k\}$. Iterate on R' until all nodes are paired. This algorithm is consistent: the probability of correct identification converges to 1 as the number of probes grows; see [2]. Several adaptations of this approach can be made to infer general (i.e. non-binary) trees. The simplest is to use the binary algorithm described above, and then to transform the resulting binary tree by pruning, i.e., removing and joining the endpoints of each link with inferred loss rate less than some threshold ε.

General Grouping Algorithms The above approach can be extended by replacing shared loss with any function on the nodes that (i) increases on moving further from the source; and (ii) whose value at a given node can be consistently estimated from measurements at receivers descended from that node. Since the mean and variance of the cumulative delay from the source to a given node have the above properties, multicast end-to-end delay measurements can also be used to infer the multicast topology

Direct Maximum Likelihood Classification In the direct MLE approach, for each possible topology we calculate the maximal likelihood of the measured outcomes as the link probabilities α_k are varied. We then maximize this over all topologies; the maximizing topology is our estimate. This classifier is consistent [2].

Accuracy and Comparison Experiments show similar accuracy for all the approaches described above. However, computational costs differ widely. If implemented as an exhaustive search through all possible trees, the cost of the direct MLE classifier grows rapidly with the number of receivers. Grouping methods avoid this since each grouping narrows the set of viable topologies. Amongst all methods considered, the binary grouping method has near optimal accuracy and is simplest to implement.

3 Experimental Results

In this section we briefly describe our efforts to validate the MINC methodology. We first describe, in Section 3.1, the results of a measurement study in which we collected end-to-end loss traces from the MBone and validated the results from inferences of loss rates collected using the Internet tool `mtrace`. We then describe, in Section 3.2, the results from more detailed simulation studies of both loss and delay.

3.1 Measurement Experiments

To validate MINC under real network conditions, we performed a number of measurement experiments on the MBone, the multicast-capable subset of the Internet. Across our experiments we varied the multicast sources and receivers, the time of day, and the day of the week. We compared inferred loss rates to directly measured loss rates for all links in the resulting multicast trees. The two sets of quantities agreed closely throughout.

During each experiment, a source sent a stream of sequenced packets to a collection of receivers over the course of one hour. It sent one 40-byte packet every 100 milliseconds to a specific multicast group. The resulting traffic stream placed less than 4 Kbps of load on any one MBone link. At each receiver, we made two sets of measurements on this traffic stream using the `mtrace`(see [8] for a description) and `mbat` software tools.

We used `mtrace` to determine the topology of the multicast tree. `mtrace` traces the *reverse* path from a multicast source to a receiver. It runs at the receiver and issues trace queries that travel hop by hop along the multicast tree towards the source. Each router along the path responds to these queries with its own IP address. We determined the tree topology by combining this path information for all receivers.

We also used `mtrace` to measure per-link packet losses. Routers also respond to `mtrace` queries with a count of how many packets they have seen on the specified multicast group. `mtrace` calculates packet losses on a link by comparing the packet counts returned by the two routers at either end of the link. We ran `mtrace` every two minutes to obtain thirty separate loss measurements during each one-hour experiment. We also verified that the topology remained constant during our experiments by inspecting the path information we obtained every two minutes.

It is important to note that `mtrace` does not scale to measurements of large multicast groups if used in parallel from all receivers as we describe here. Parallel `mtrace` queries converge as they travel up the tree. Enough such queries will overload routers and links with measurement traffic. We used `mtrace` in this way only to validate MINC on relatively small

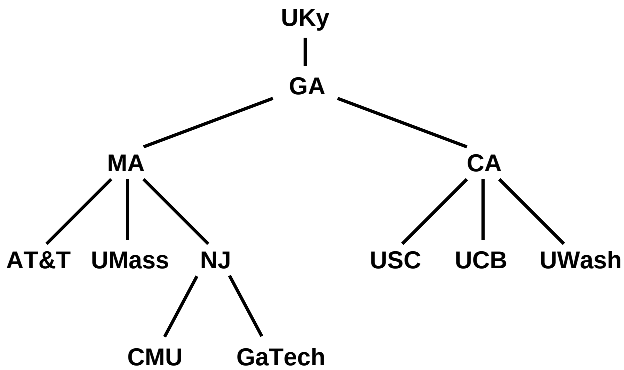

Figure 2: Multicast routing tree during our representative MBone experiment.

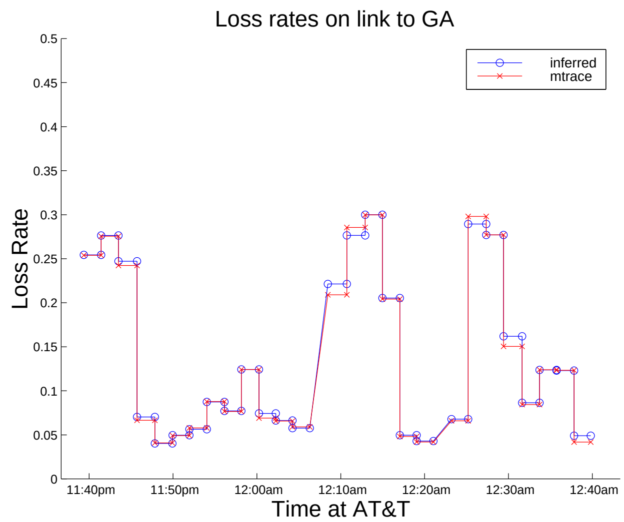

Figure 3: Inferred vs. actual loss rates on link between UKy and GA.

multicast groups before we move on to use MINC alone on larger groups.

We used the `mbat` tool to collect traces of end-to-end packet losses. `mbat` runs at a receiver, subscribes to a specified multicast group, and records the sequence number and arrival time of each incoming packet. We ran `mbat` at each receiver for the duration of each hour-long experiment.

We then segmented the `mbat` traces into two-minute subtraces corresponding to the two-minute intervals on which we collected `mtrace` measurements. Finally, we ran our loss inference algorithm on each two-minute interval and compared the inferred loss rates with the directly measured loss rates.

Here we highlight results from a representative experiment on August 26, 1998. Figure 2 shows the multicast routing tree in effect during the experiment. The source was at the U. of Kentucky and the receivers were at AT&T Labs, U. of Massachusetts, Carnegie Mellon U., Georgia Tech, U. of Southern California, U. of California at Berkeley, and U. of Washington. The four branch routers were in California, Georgia, Massachusetts, and New Jersey.

Figure 3 shows that inferred and directly measured loss rates agreed closely despite a link experiencing a wide range of loss rates over the course of a one-hour experiment. Each short horizontal segment in the graph represents one two-minute, 1,200-probe measurement interval. As shown, loss rates on the link between the U. of Kentucky and Georgia varied between 4% and 30%. Nevertheless, differences between inferred and directly measured loss rates remained below 1.5%.

In summary, our MBone experiments showed that inferred and directly measured loss rates agreed closely under a variety of real network conditions:

- Across a wide range of loss rates (4%–30%) on the same link.
- Across links with very low ($<$ 1%) and very high ($>$ 30%) loss rates.
- Across all links in a multicast tree regardless of their position in the tree.
- Across different multicast trees.
- Across time of day and day of the week.

Furthermore, in all cases the inference algorithm converged to the desired loss rates well within each two-minute, 1,200-probe measurement interval.

3.2 Simulation Experiments

We have performed more extensive validations of our inference techniques through simulation in two different settings: the simulation of the model with Bernoulli losses and simulations of networks with realistic traffic. In the model simulations, probe loss and delay obey the independence assumption of the model. We applied the inference algorithm to

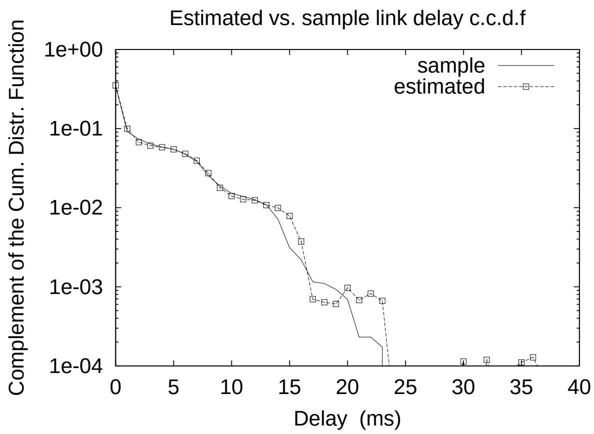

Figure 4: Inferred and Sample Delay ccdf. for a leaf link in the topology of Figure 2.

the end-to-end measurements and compared the inferred and actual model parameters for a large set of topologies and parameter values. We found that loss rates, mean delay, and variance estimates converged to close to their actual values with 2,000 probes. The number of probes required to accurately compute the entire delay distributions is higher. In our experiments we found good agreement with 10,000 probes.

The second type of experiment is based on the `ns` simulator. Here delay and loss are determined by queueing delay and queue overflow at network nodes as multicast probes compete with traffic generated by TCP/UDP traffic sources. Multicast probe packets are generated by the source with fixed mean interarrival times; we used CBR or Poisson probes. We simulated different topologies with different background traffic mixes comprising infinite FTP sessions over TCP and exponential or Pareto on-off UDP sources. We considered both Drop Tail and Random Early Detection (RED) buffer discard methods, [5].

We compared the inferred loss and delay with actual probe loss and delay. We found rapid convergence of the estimates although with small persistent differences. We attribute this to the presence of spatial dependence, i.e., dependence between probe losses and delays on different links. This can arise through correlations in the background traffic due to correlation arising from TCP dynamics, e.g., synchronization between flows as a result of slow-start after packet loss. We have shown in [1] that small deviations from the spatial independence assumption lead to only small errors in inference.

We also found that background traffic introduces temporal dependence in probe behavior, e.g., its burstiness can cause back-to-back probe losses. We have shown that while temporal dependence can decrease the rate of convergence of the estimators, consistency is unaffected. In the experiments the inferred values converged within 2,000 probes despite the presence of temporal dependence.

While there is understanding of mechanisms by which temporal and spatial dependence can occur, as far as we know there are no experimental results concerning its magnitude. We believe that large or long lasting dependence is unlikely in the Internet because of traffic and link diversity. Moreover, we expect loss correlation to be reduced by the introduction of RED.

We also compared the inferred probe loss rates with the background loss rates. The experiments showed these to be quite close, although not as close as inferred and actual probe loss rates. We attribute this to the inherent difference in the statistical properties of probe traffic and background traffic, principally due to TCP traffic being more bursty than probe traffic and to TCP adapting its sending rate when it detects losses.

To illustrate the distribution of delay inference results, we simulated the topology of the multicast routing tree shown in Figure 2. In order to capture the heterogeneity between edges and core of a network, interior links have higher capacity (5Mb/sec) and propagation delay (50ms) than those at the edge (1Mb/sec and 10ms). Background traffic comprises infinite FTP sessions and exponential on-off UDP sources. Each link is modeled as a FIFO queue with a 4-packet capacity. Real buffers are usually much larger; the capacity of four is used to reduce the time required to simulate the network. The discard policy is Drop Tail. In Figure 4, we plot the inferred versus the sample complementary cumulative distribution function (discretized in one millisecond bins) for one of the leaf links, using about 18,000 Poisson probes. The estimated distribution closely follows the sample distribution and is quite accurate for tail probabilities greater than 10^{-2}. Note that the estimated distribution is not always monotonically decreasing. This is because negative probabilities are occasionally estimated in the tail due to an insufficient number of samples. It is worth pointing out that, given the irregular shape of the sample distribution, the same level of accuracy would not be possible using a parametric model. We also observed in these experiments that the inferred distributions are less accurate for the higher capacity interior links. This appears to be caused by the difference in delay range among different links that negatively affects those with relatively smaller delays. In the presence of high spatial correlation (up to $0.4 \sim 0.5$) the inferred tail distribution can diverge from the actual one. However, the delay mean and variance are insensitive to this tail behavior.

4 Deployment Efforts

We have observed in the previous section that MINC is a very promising methodology for providing detailed internal network performance characteristics. In this section we describe our efforts in deploying this methodology and making it available on the Internet. Our efforts are threefold. First, we are continuing the development of NIMI to support multicast-based measurement experiments. This is described in Section 4.1. Second, we have identified the real-time transport protocol, RTP, and its associated control protocol, RTCP, as promising mechanisms for generating and collecting end-to-end multicast measurement traces. Our efforts in developing an RTP-based tool are described in Section 4.2. Last, Section 4.3 contains a description of an analysis and visualization tool, MINT (Multicast Inference Network Tool), that is currently under development.

4.1 Deployment on NIMI

A major difficulty with characterizing Internet dynamics comes from the network's immense heterogeneity [12]. Load patterns, congestion levels, link bandwidths, loss rates, protocol mixes, the patterns of use of particular protocols—all of these exhibit great variation both at different points in the network, and over time as the network evolves. Accordingly, to soundly characterize some aspects of Internet behavior, requires measuring a diverse collection of network paths. It is not adequate to measure between just a few points, regardless of how carefully done.

The same problem arises in assessing the accuracy of measurement techniques such as MINC. To address this concern, we are deploying MINC measurement utilities within the National Internet Measurement Infrastructure (NIMI) [10]. NIMI consists of a number of measurement "platforms" deployed at various locations around the Internet. Each platform is capable of sourcing and sinking active measurement traffic and recording the timing of the traffic at both sender and receiver. Measurement "clients" that wish to use the infrastructure make authenticated requests to the platforms to schedule future measurement activity.

A key property of such an infrastructure is its N^2 scaling: if the infrastructure consists of N platforms, then they together can measure network traffic along $O(N^2)$ distinct paths through the network. Consequently, with a fairly modest N, one can obtain a wide cross-section of the network's diverse behavior. (The NIMI infrastructure currently consists of 31 sites, expected to grow to 50 by summer 2000).

Using NIMI for MINC measurements required several extensions to NIMI. The first was modifying the standard NIMI packet generator, *zing*, to send and receive multicast traffic, and the corresponding analysis program, *natalie*, to incorporate the notion that a single packet might arrive at several places (and fail to arrive at others). MINC has also required the generalization of NIMI control mechanisms, in order to allow for a single measurement run spanning multiple senders and receivers. A possible further step would be to use multicast itself for both scheduling measurements and disseminating the results. But this change to NIMI remains for future work, as it raises the thorny problem of devising a scalable-yet-reliable multicast transport protocol.

Our experiences with using NIMI to date have been quite frustrating, not due to the infrastructure itself, but because of the poor quality of multicast connectivity between the different platforms. Until recently, at best only 1/3 of the NIMI platforms had multicast connectivity between them. We gather anecdotally that problems with poor interdomain multicast connectivity have been endemic to the Internet. Recently, connectivity has begun to improve, and it appears likely that over the next several years it will continue to do so, as agreement is reached on the proper set of intradomain and interdomain routing protocols and the interoperation between them. We are also attempting to address this problem in two ways: (1) to grow the NIMI infrastructure by adding sites with high-quality multicast connectivity; and (2) to investigate theoretical work on inferring network characteristics using correlated *unicast* traffic, where, instead of exploiting the perfect correlations inherent in multicast packet reception, we send back-to-back unicast packets and attempt to exploit the considerably weaker correlations in their loss and delay patterns.

4.2 Integration with RTCP

We are developing tools to apply MINC in real-time, so that MINC could be used by applications to respond to changing network conditions in new and more sophisticated ways. For example, a management program might adaptively adjust its probes to home in on a problem router.

Our tools transmit network information using RTCP, the control protocol for multicast transport protocol RTP [14]. By sharing their traces using RTCP, they benefit from RTCP's built-in scaling mechanisms.

The approach is based on three tools: `mgen`, `mflect`, and `mmerge` (Figure 5). `mgen` generates a stream of data (and may be replaced by any other application that multicasts data over RTP). A copy of `mflect` at each receiver maintains loss traces for the packets it does and does not receive from `mgen`. It periodically multicasts these (in a sense reflecting the data stream: hence "mflect"). `mmerge` collects the traces sent by `mflect`, collating those from the different data receivers and making them available to a tool, such as MINT, for inference.

`mflect` and `mmerge` are designed so that they may be incorporated directly into existing and future multicast applications. Their joint functionality is available as an extension to the RTP common code library from University College London, called RTPXR, ("eXtended Reporting"). An application using RTPXR would be in a position to respond adaptively to information on the topology of its data distribution tree.

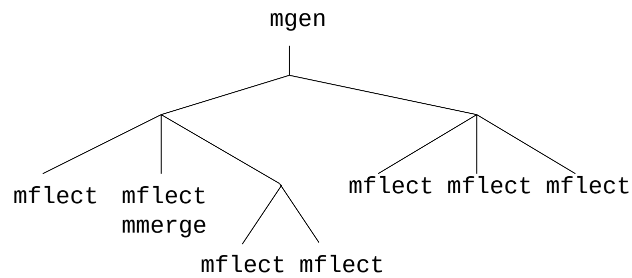

Figure 5: An RTCP-based tool deployment example, on the same topology as shown in Figure 2, with inference being performed at UMass.

Ongoing research related to these tools concerns the scalability of trace sharing. For example, a raw bit vector loss trace for 3,000 packets would consume 375 octets, far more than the four octets allocated for summary loss information in a standard RTCP packet. To limit the traces to an acceptable intermediate size we are investigating the use of compression techniques such as run length encoding, as well as distributed methods by which all copies of `mflect` in a single session can agree on which portions of the trace to share in place of the whole trace.

4.3 MINT: Multicast Inference Network Tool

MINT is intended to facilitate multicast-based inference. It takes as inputs all of the traces collected from the end-hosts. These traces may or may not include `mtrace` outputs. Currently, MINT comprises three components: a graphical user interface (GUI), a topology discovery algorithm, and an inference engine. Users interact with the GUI to manipulate the inference such as choosing number of samples, visualizing the multicast tree with losses or showing the performance evolution over specific links. Depending on the availability of mtrace output, MINT discovers the topology by either parsing mtrace inputs or inferring the multicast tree from the loss traces. The inference engine takes topology information and loss traces to infer the network internal loss and then provides this to the GUI. The user can then view the results in one of several ways. One way is to lay out the logical multicast tree and display the links in

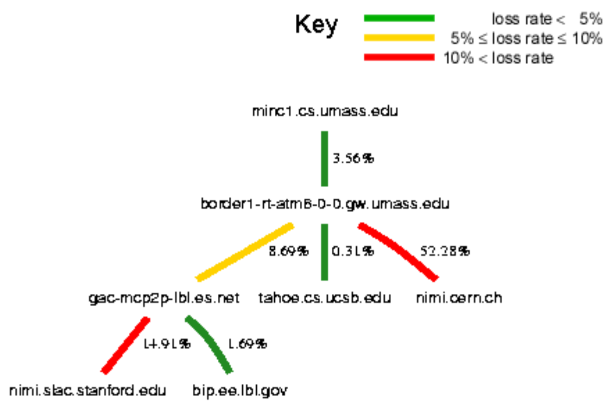

Figure 6: MINT view of the logical multicast tree with losses.

different colors to distinguish different average loss rates (e.g., Figure 6). The user can also focus on a single link and observe how the loss rate evolves over time for that link.

Our future plans for MINT are to include support for delay inference, to test it thoroughly by feeding it with daily traces collected from NIMI, and to make its output available to the community on the WOrld WIde Web.

5 Related Work

A growing number of measurement infrastructure projects (e.g., AMP, Felix, IPMA, NIMI, Surveyor, and Test Traffic [3]) aim to collect and analyze end-to-end performance data for a mesh of unicast paths between a set of participating hosts. We believe our multicast-based inference techniques would be a valuable addition to these measurement platforms. We are continuing to work on incorporating MINC capabilities into NIMI.

Recent experimental work has sought to understand internal network behavior from endpoint performance measurements (e.g., TReno [9]). In particular, `pathchar` [11] is under evaluation as a tool for inferring link-level statistics from end-to-end unicast measurements. Much work remains to be done in this area; MINC contributes a novel multicast-based methodology.

Regarding multicast-based measurements, we have already described the `mtrace` tool. This forms the basis for several tools for performing topology discovery (`tracer` [6]) and visualizing loss on the multicast distribution tree of an application (MHealth [8]). However, `mtrace` suffers from performance and applicability problems in the context of large-scale Internet measurements. First, `mtrace` needs to run once for each receiver in order to cover a complete multicast tree, which does not scale well to large numbers of receivers. In contrast, MINC covers the complete tree in a single pass. Second, `mtrace` relies on multicast routers to respond to explicit measurement queries. Although current routers support these queries, providers may choose to disable this feature since it gives anyone access to detailed delay and loss information about paths inside their networks. In contrast, MINC does not rely on cooperation from any internal network elements.

6 Conclusions

We have described a new approach to identifying internal network characteristics based on the use of end-to-end multicast measurements. This methodology is rigorously based in estimation theory. A preliminary evaluation for identifying loss rates based on measurements made over the MBone indicates that it is accurate and readily able to track dynamic fluctuations that occur over time. More detailed investigations based on simulation further corroborate this conclusion, not only for the case of losses, but delays as well. Finally, we described our current efforts to deploy this methodology on the Internet and make it available to the community at large.

We believe that MINC is an important new methodology for network measurement, particularly Internet measurement. It does not rely on network cooperation and it should scale to very large networks. MINC is firmly grounded in statistical analysis backed up by packet-level simulations and now experiments under real network conditions. We are continuing to extend MINC along both analytical and experimental fronts.

Acknowledgements

Mark Handley suggested using RTCP receiver reports to carry MINC loss traces.

References

[1] R. Cáceres, N.G. Duffield, J. Horowitz, D. Towsley, "Multicast-Based Inference of Network-Internal Loss Characteristics," *IEEE Transactions on Information Theory*, Nov. 1999.

[2] R. Cáceres, N.G. Duffield, J. Horowitz, F. Lo Presti, D. Towsley, "Loss-Based Inference of Multicast Network Topology," *Proc. 1999 IEEE Conf. on Decision and Control*, Phoeniz, AZ, December 1999.

[3] Cooperative Association for Internet Data Analysis, "Internet Measurement Efforts," `http://www.caida.org/Tools /taxonomy.html/InternetMeasurement`, 1999.

[4] N.G. Duffield, F. Lo Presti, "Multicast Inference of Packet Delay Variance at Interior Networks Links", *Proc. Infocom'2000*, Tel Aviv, Israel, March 26–30, 2000.

[5] S. Floyd, V. Jacobson. "Reandom Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. on Networking*, 1(4):397–413, Aug. 1993.

[6] B.N. Levine, S. Paul, J.J. Garcia-Luna-Aceves, "Organizing Multicast Receivers Deterministically According to Packet-Loss Correlation," *Proc. ACM Multimedia 98*, September 1998.

[7] F. Lo Presti, N.G. Duffield, J, Horowitz, D. Towsley. "Multicast-Based Inference of Network-Internal Delay Distributions," preprint, AT&T Labs and University of Massachusetts, 1999.

[8] D. Makofske and K. Almeroth, "MHealth: A Real-Time Multicast Tree Visualization and Monitoring Tool", *Proc. NOSSDAV'99*, Basking Ridge New Jersey, USA, June 1999.

[9] M. Mathis, J. Mahdavi, "Diagnosing Internet Congestion with a Transport Layer Performance Tool," *Proc. INET '96*, June 1996.

[10] V. Paxson, J. Mahdavi, A. Adams, M. Mathis, "An Architecture for Large-Scale Internet Measurement," *IEEE Communications*, 36(8):48–54, August 1998.

[11] A. Downey, "Using pathchar to estimate Internet link characteristics", *Proc. ACM SIGCOMM '99*, September 1999.

[12] V. Paxson and S. Floyd, "Why We Don't Know How to Simulate the Internet," *Proc. 1997 Winter Simulation Conference*, Atlanta, GA, December 1997.

[13] S. Ratnasamy, S. McCanne, "Inference of Multicast Routing Tree Topologies and Bottleneck Bandwidths Using End-to-End Measurements," *Proc. IEEE Infocom '99*, March 1999.

[14] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 1889, January 1996.