Inference and Labeling of Metric-Induced Network Topologies

Azer Bestavros, John Byers, Khaled Harfoush

Abstract— **The development and deployment of distributed network-aware applications and services require the ability to compile and maintain a model of the underlying network resources with respect to (one or more) characteristic properties of interest. To be manageable, such models must be compact, and to be general-purpose, they should enable a representation of properties along temporal, spatial, and measurement resolution dimensions. In this paper, we propose MINT—a general framework for the construction of such metric-induced models using end-to-end measurements. We present the basic theoretical underpinnings of MINT for a broad class of metrics obeying certain properties. We instantiate MINT for two metrics of interest, namely packet loss rates and bottleneck bandwidth. For the loss rate metric, we leverage recently proposed end-to-end techniques for the estimation of shared losses to characterize loss topologies. We present results of simulations and Internet measurements that confirm the effectiveness and robustness of our loss topology constructions over a wide range of network conditions.**

I. Introduction

Many of today's Internet services are administered through co-location facilities that are distributed over the wide area. Examples include Content Distribution Networks (CDNs) and Application Service Providers (ASPs). Internet servers at such facilities typically command a large number of concurrent unicast connections. We use the term *Mass Servers* to refer to such *Mass*ively Accessed *Servers*.

A collection of Mass servers distributed over a WAN is in a unique position to infer valuable network state information that could be effectively used to maximize resource utilization and optimize content delivery and distribution. In order for such *network-aware strategies* to be practical, an endpoint must be able to quickly and accurately infer the conditions of network resources connecting it to other endpoints. Typically, these conditions are measured in terms of one or more metrics of interest, such as available bandwidth, hop-count, loss rate, delay, and jitter.

Previous studies aimed at the identification of network-internal conditions (whether static or dynamic) have focused on the characterization of physical network entities (e.g., routers, links, AS boundaries). For the resource management problems that face Mass servers, an accurate characterization of all physical resources between the server and its clients is not necessary, nor feasible. It is overkill. Rather, an abstraction that captures the conditions of only those shared resources to be judiciously managed is sufficient. For a streaming media delivery application, a model of the network that captures path *bottleneck bandwidth* and *jitter* may be used for server selection purposes. Alternatively, for a distributed caching or content distribution application, a model of the network that captures *loss rates* on congested paths may be used for optimizing replica placement.

End-to-end measurements made in the course of normal operations by a Mass server provide a wealth of information about the end-to-end characteristics of a particular path in the network. For example, end-to-end bottleneck bandwidth rates, round-trip times and packet loss statistics can all be inferred from the dynamics of a TCP connection [1]. In addition to these connection-specific statistics, end-to-end measurements correlated across different connections (at one or more Mass servers) can be used to infer conditions at a level that is more granular than that encompassing an entire path, and yet less granular than that at a specific network resource (e.g., for a portion of a path as opposed to for every hop along the path). This paper proposes a framework for achieving this efficiently for a wide range of metrics. In particular, it proposes a framework that enables a parametrized abstraction (or summarization) of the topology connecting a set of endpoints with respect to a given metric of interest.

Metric-Induced Network Topologies: Given a set of network endpoints, we define a *Metric-Induced Network Topology* (MINT) to be a weighted, directed graph. The vertices of this graph represent network endpoints as well as routers, while an edge in this graph represents a *sequence* of one or more physical network links. The weight on each edge corresponds to a real-valued quantity that a specific metric function attributes to the sequence of physical links represented by the edge. Only those sequences of links which collectively have a metric value above a threshold $c \geq 0$ are represented by edges in this topology. The threshold c acts as a "compression parameter"—the larger the value of c, the less granular (i.e. less detailed) the resulting MINT graph will be. By hiding uninteresting details of a physical topology (e.g. sequences of links with abundant bandwidth, negligible jitter, or insignificant losses), MINT graphs enable a compact representation of those resources that need to be accounted for by network-aware applications at Mass servers.

Paper Contributions and Overview: Existing techniques for making inferences from end-to-end observations are special-purpose, each targeting specific metrics of interest, such as estimation of bottleneck bandwidth, packet loss rate, propagation delay or jitter.

In this paper, we study the extent to which such end-to-end methods can be effective, not by considering individual specialized techniques, but by reasoning about properties of the metrics themselves.

The authors are with the Computer Science Department at Boston University. E-mail: {best,byers,harfoush}@cs.bu.edu. This work was partially supported by NSF research grants ANI-9986397, ANI-0095988 and CAREER ANI-0093296.

Our work introduces the concept of Metric-Induced Network Topologies (MINT) and proposes general procedures for the inference and labeling of MINT topologies, as well as the integration of MINT topologies obtained at different points in time and from different network vantage points. We specify a broad class of metrics for which MINT procedures are applicable, focusing on two specific metrics defined over a network path—namely, packet loss rate and bottleneck bandwidth.

For the loss rate metric, we demonstrate that recently proposed end-to-end techniques for the estimation of shared losses can be leveraged to enable the characterization of *loss topologies*. We present results both of extensive simulations and of Internet deployment results that demonstrate the accuracy and convergence of our loss topology inference and labeling techniques.

II. Related Work

A Taxonomy: One widely adopted strategy for the characterization of network properties/conditions is to analyze data collected from network-internal resources (e.g. router ICMP replies, BGP routing tables) to generate performance reports [15], [20], [16] or to perform Internet topology characterization [10], [9], [22]. This approach is best applied over long time scales to produce aggregated analyses, but does not lend itself well to providing answers to the fine-grained needs of network-aware applications.

Another approach, applicable at shorter time scales, is statistical inference of network internal characteristics based on end-to-end measurements of point-to-point traffic. Although this strategy has been employed in networking contexts for at least a decade, starting with the use of packet-pair probes [17], [3], recent efforts have greatly expanded the set of available techniques [27], [24], [23], [26].

The diversity of inference techniques cut across approaches which are sender-driven, receiver-driven, those which employ multicast and those which employ unicast probes, and span metrics ranging from static metrics (e.g. bottleneck bandwidth) to dynamic metrics (e.g. packet loss rates). However, none of these studies attempt to provide a *general* framework to describe when inferences may be drawn. Our approach does so, by focusing not on the specific probing technique, but by concentrating on the *metrics* of interest, and by developing an understanding of which metrics are amenable to inference techniques.

Estimation of Shared Loss Using End-to-End Measurements: The specific problem of inferring and labelling loss topologies is motivated in part by recent work on topological inference over *multicast* sessions [4], [7], [24]. By making purely end-to-end observations of packet loss at endpoints of multicast sessions, Ratnasamy and McCanne [24] and Cáceres *et al.* [4] have demonstrated how to make unbiased, maximum likelihood estimation inferences of (a) the multicast tree topology and (b) the packet loss rates on the edges of the tree, respectively.

At the core of our methodology for constructing loss topologies is the need for a procedure for the measurement of shared losses between one source and multiple destinations.

A number of recent efforts have addressed this problem; all would be suitable as underlying procedures in our framework. In [25], Rubenstein, Kurose, and Towsley propose the use of end-to-end probing to detect shared points of congestion (POCs). By their definition, a point of congestion is shared when a set of routers are dropping and/or delaying packets from both flows. Their probing technique for identifying POCs uses a Poisson process to generate probe traffic to a pair of remote endpoints and computes cross-correlation measures between pairs of packets from these flows. In [12], we presented an alternative technique for the identification of shared losses using a Bayesian Probing (BP) approach. Using BP, packet-pair probes are sent to a pair of *different* receivers to introduce loss and delay correlation, much the same way a multicast packet to these two receivers introduces correlation. Other similar techniques have been recently proposed [8], [5], [18], [19]. For example, the striped unicast probing of Duffield, LoPresti, Paxson, and Towsley uses a sequence of back-to-back packets sent to different receivers as an approximation of a multicast probe, thus enabling them to use link loss and delay inference techniques devised for multicast probes [7].

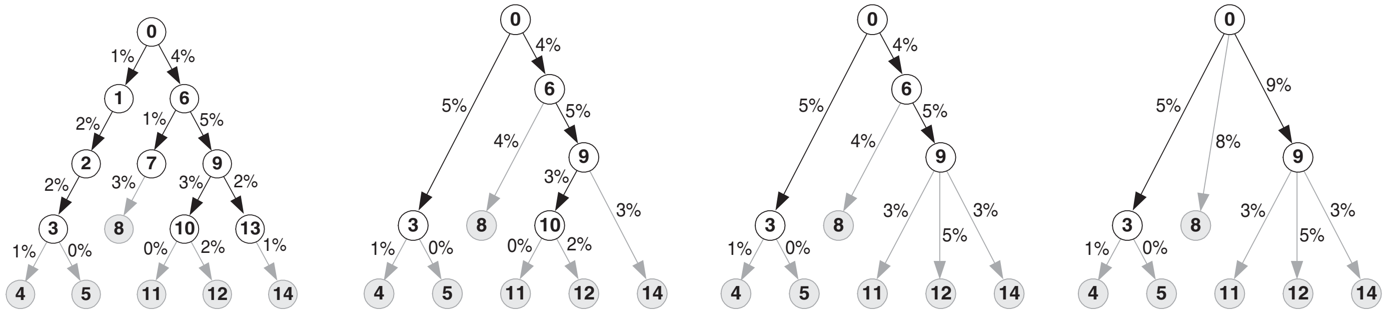
III. Metric-Induced Network Topologies

As motivated in the introduction, for many applications, it is often convenient to cluster clients of a Mass Server according to the extent to which they consume a shared network resource. Such sharing is often quantified by one of a number of *metrics*, including shared jitter, shared network delay or shared packet loss.

A. Basic Definitions

The labeled topology from the server to the clients with respect to a given performance metric is a *Metric-Induced Network Topology* (MINT). In many instances, edges with small labels, representing negligible, or statistically insignificant amounts of delay, jitter or loss, can safely be disregarded. Therefore, effective methods for producing a metric-induced topology must be *sensitive* to different possible thresholds for what constitutes an observable value of the metric. In this sense, the MINT framework provides a multiscale representation of a metric-induced topology. Beginning with basic definitions used also in prior work, we formalize these notions below.

Consider the set of links used to route unicast traffic between a server and a set of clients. Together these links form a tree $\mathcal{T} = (V, E)$ rooted at the server, with the clients at the leaves and routers at the internal nodes. The flows of packets sent from a server to an arbitrary subset of its clients share some (possibly none) of $\mathcal{T}$'s links and eventually diverge on separate links en route to the different clients.



(a) Physical tree

(b) Loss tree ($c = 0$)

(c) Loss tree ($0.03 < c \leq 0.04$)

(d) Loss tree ($0.04 < c \leq 0.05$)

Fig. 1. Relationship between physical and loss topologies under various sensitivity parameter (c).

Definition 1: The *physical topology* connecting a server to a set of clients is the tree $\mathcal{T}$ induced by IP routing with the server at the root, clients at the leaves and routers at the internal nodes of the tree.

The following operation is needed to help us define the portion of a physical topology observable through end-to-end measurements. We say that we *collapse* a node i of into its parent j if we delete edge (i, j) from the tree and attach all children of i to j, by replacing all edges (i, k) with edges (j, k) for all $k \neq j$. Note that when this operation is applied to a tree, the resulting topology is also a tree.

Definition 2: The *logical topology* induced by a physical topology $\mathcal{T}$ is formed from $\mathcal{T}$ after all internal nodes with only one child have been collapsed into their parent recursively.

A logical topology reflects the topology made visible by end-to-end measurements, as end-to-end techniques will be unable to assign metrical labels on a link-by-link basis to a sequence of physical links in a chain with no branching.

We now extend our definitions to apply to topologies with edges labeled according to a metric. For the purpose of our discussion, we define a *metric* to be a function f whose domain is the set of paths in a tree (or set of simple paths in a graph) and whose range is the reals. We refer to a *labeled topology* as any topology in which values of the metric have been applied to each link in the topology, noting that links may correspond to multi-hop physical paths as well as physical links.

For some metrics, end-to-end observations will have difficulty distinguishing a small metrical value ϵ from zero and as a result, may misclassify incidence of sharing. For this reason, we parameterize any labeling algorithm with a sensitivity parameter c which is the minimum value of a label that can be applied to any internal link in the resulting topology $\mathcal{T}_c$. Our definition of metric-induced topology incorporates such a parameter:

Definition 3: A metric-induced topology with sensitivity parameter c is the labelled topology formed when all internal nodes i in a logical topology whose parent link L_i has a value $f(L_i) \leq c$ have been collapsed into their parent.[1]

Based on these definitions, an edge in a MINT graph represents a sequence of one or more physical network hops that collectively exhibit observable values of that metric. Clearly, the larger the value of the sensitivity parameter c, the smaller the number of links present in the topology $\mathcal{T}_c$. In this sense, $\mathcal{T}_c$ represents a "condensed" version of the original topology, and increasing the value of the sensitivity parameter c has the effect of increasing the level of condensation.

We refer to the metric-induced topology for which $c = 0$ (i.e. $\mathcal{T}_0$) as the *base topology*, while a metric-induced topology with $c > 0$ can result in more extensive condensation, in which arbitrary connected subgraphs of internal nodes may collapse into a single node. The following example illustrates this behavior for the specific case of loss topologies.

Consider the physical tree topology shown in Figure 1(a). Node 0 is the server, nodes 4, 5, 8, 11, 12, and 14 are the clients and the remaining nodes are routers. The links in Figure 1(a) are labeled with the actual loss rates on these links. Figure 1(b) shows the loss topology $\mathcal{T}_0$ for this physical tree when $c = 0$. Notice that in $\mathcal{T}_0$, paths with intermediate nodes of unit out-degree (e.g. the path between nodes 0 and 3 and the path between nodes 6 and 8) are collapsed into a single (logical) link. Figures 1(c) and 1(d) show the loss topologies for this physical tree when $0.03 < c \leq 0.04$ and when $0.04 < c \leq 0.05$, respectively. These topologies are obtained from T_0 by collapsing internal links with loss probabilities that are less than the sensitivity parameter c. For the sake of simplicity in this example, we have taken the liberty of merging loss probabilities on consecutive links by using simple addition (at the cost of a small degree of inaccuracy).

B. MINT Inference and Labeling

The topology inference framework we present next applies to a broad class of metrics (encompassing packet loss rates, propagation delay, and network bandwidth) satisfying three technical conditions. We will define those conditions next, then go on to demonstrate how our inference techniques can be (1) recursively applied to large topologies, (2) applied to incorporate

[1] The values on adjusted links must be updated in a metric-specific way after each collapse operation. We discuss the significance of this in Section III-B.

observations taken at multiple points in time, and (3) applied to incorporate observations taken at multiple points in space.

To define these conditions, we need a bit more terminology. We denote a path between 2 hosts A and B by $p_{A \to B}$ and denote a general path between any two end hosts as p_i, where the subscript represents a path identifier. We refer to path p_i as a *subpath* of p_j if the sequence of edges forming p_i is a subsequence of the edges forming p_j. Let us then denote $p_{i.j}$ as the maximal subpath common to both p_i and p_j, i.e. $p_{i.j}$ is the shared portion of p_i and p_j.

We classify metrics based on three different properties:

Monotonicity: A metric f is *monotonically non-decreasing* (*non-increasing*) if for any pair of paths p_i, p_j for which p_i is a subpath of p_j, $f(p_i) \leq f(p_j)$ ($f(p_i) \geq f(p_j)$).

Separability: A metric f is *separable* if for any path p composed of the union of two disjoint subpaths p_i and p_j, $f(p_j) = g(f(p), f(p_i))$ for some function g.

Symmetry: A metric f is *symmetric* if for any path p_i the value $f(p_i)$ is equal as observed from either end of p_i.

Metrics of interest are diverse with respect to these three properties. For example, the loss rate metric is monotone, separable but not symmetric; propagation delay is monotone, separable and (often) symmetric; available bandwidth is monotone, but not necessarily symmetric and definitely not separable; and jitter is neither monotone, nor separable, nor symmetric.

We will now show how the presence (or absence) of each these properties in a metric f affect our ability to infer, label and aggregate topologies induced by f using end-to-end measurements. We do so through a sequence of theorems. Complete proofs of these theorems are available in an expanded version of this paper [2]. Algorithmic steps used in the sketch of Theorem 1 are representative of those required to prove the other theorems, and are similar to those discussed in [6].

Theorem 1: Given a procedure that enables the evaluation of $f(p_a)$, $f(p_b)$, and $f(p_{a.b})$ for some monotone metric f between a source s_0 and any two endpoints a and b, one can efficiently *infer* the base topology induced by f over an arbitrary physical topology $\mathcal{T}$. In the event that f is separable, one can also efficiently *label* the topology induced by f for any sensitivity parameter $c > 0$.

Proof: (Sketch). We first demonstrate how to recursively infer the base topology based on end-to-end evaluations of a monotone function f. Consider a set of n endpoints $s_1, s_2, \ldots s_n$. Sort all pairs of endpoints (s_u, s_v) sorting on the value of $f(p_{s_u.s_v})$. Let s_i and s_j be the pair[2] of endpoints for which $f(p_{s_i.s_j})$ is maximum.

Reduction: Monotonicity ensures that path $p_{s_i.s_j}$ cannot be a subpath of any other path connecting the source s_0 to any endpoint other than s_i and s_j. Thus there exists an internal node r at which the paths from s_0 to s_i and s_j diverge; moreover, r is *not* on any path connecting s_0 to any endpoint other than i and j. The subtree rooted at r is therefore completely specified.

Recursion: The above construction identifies an internal node r in the logical topology, and gives a method for computing $f(p_r) = f(p_{s_i.s_j})$. This construction therefore allows us to remove s_i and s_j and replace them with r, thereby reducing the problem of finding the logical topology connecting a server s_0 to n endpoints to the smaller problem of finding the logical topology connecting that same server to strictly fewer than n endpoints. Applying this reduction recursively, one can produce the base topology with respect to f.

In the event that the metric f is also separable, we can prove the second part of Theorem 1 for any sensitivity parameter $c > 0$ by repeatedly applying the following step:

Compression and Relabeling: In a bottom-up fashion, we collapse an *internal* node r_i into its parent node r_j iff $g(f(p_{r_i}), f(p_{r_j})) < c$. Recall that collapsing r_i into r_j entails deleting edge (r_i, r_j) from the tree and attaching all children of r_i to r_j. The label of the edges connecting former children of r_i to r_j must be updated; by the separability condition, that is by an amount equal to $g(f(p_{r_i}), f(p_{r_j}))$. The result is a labeled topology as depicted in Figure 1. ■

Theorem 2: Given a procedure that enables the evaluation of $f(p_{s_i \to a})$, $f(p_{s_j \to a})$, and $f(p_{s_i \to a.s_j \to a})$ for some monotone metric f between any two sources s_i and s_j and an endpoint a, one can efficiently *infer* the base topology induced by f over an arbitrary physical topology $\mathcal{T}$ connecting any set of sources to endpoint a. In the event that f is separable, one can also efficiently *label* the topology induced by f for any sensitivity parameter $c > 0$.

C. Integrating MINT Snapshots

For non-stationary metrics (such as loss rates), it is evident that the labels placed on the edges of a metric-induced topology will change over time. Moreover, at different points in time, measured observations over a link or set of links may fail to reach the threshold specified by the sensitivity parameter. For both of these reasons, time-varying observations of a metric-induced topology measured from the same endpoints may contain different sets of observable internal edges. Therefore, one can hope to integrate a set of MINT snapshots generated at different points in time to produce an inferred (but unlabelled) topology which includes the *union* of all internal edges observed in the snapshots. It is natural to wonder whether one can produce such an integrated topology from a set of mutually consistent topological snapshots efficiently. The following theorems demonstrate that this can in fact be achieved.

[2] In general, there may be a set of nodes whose pairwise evaluations are all equal and maximum. These nodes would all be grouped together in the reduction operation.

Definition 4: Consider a set of unlabeled metric-induced tree topologies $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k$ rooted at the same source and spanning destination sets $E_1, E_2, \ldots E_k$ respectively. We say that these trees are *mutually consistent* if there exists a tree $\mathcal{T}'$ spanning all destinations in $\bigcup E_i$ from which we can generate $\mathcal{T}_i$ for any i by repeated application of the collapse operation defined earlier.

Theorem 3: The minimal tree T' spanning a set of mutually consistent tree topologies $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$ connecting a server to a set of m clients is unique and can be constructed in $O(nm^2 \log d)$ time.

Another important problem that is similar to the integration of time-varying observations is that of integrating observations made from *spatially distinct* vantage points. As motivated in the introduction, a content delivery network (CDN) consisting of a number of distributed servers already performs a vast number of end-to-end observations in the normal course of daily operations and could benefit from a methodology to integrate these observations.

We provide the following additional definitions to generalize the notions above to enable the integration of metric-induced topological snapshots made from different points in space—namely, how to produce a graph which is mutually consistent with a set of consistent snapshots collected from a set of distributed servers.

Definition 5: Consider a set of metric-induced topologies $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k$ rooted at *different* sources. We say that these topologies are *mutually consistent* if there exists a graph G spanning all endpoints in $\bigcup \mathcal{T}_i$ and when G is restricted to the routing tree induced by IP routing and spanning endpoints in $\mathcal{T}_i$, $G|\mathcal{T}_i$, $\mathcal{T}_i$ can be generated from that subgraph by repeated application of the collapse operation.

The following theorem generalizes the topology integration process to merge consistent topologies observed from an arbitrary set of sources to arbitrary subsets of clients (or between an arbitrary set of collaborating sources).

Theorem 4: Given a set of n sources $s_0, s_1, \ldots, s_{n-1}$, a set of m clients $c_0, c_1, \ldots, c_{m-1}$, and a procedure that enables the mutually consistent evaluation of:

1. $f(p_{s_i,c_k}), \forall i, k : 0 \leq i < n, 0 \leq k < m$
2. $f(p_{s_i,c_k.s_i,c_l}), \forall i, l, k : 0 \leq i < n, 0 \leq l, k < m, l \neq k$, and
3. $f(p_{s_i,c_k.s_j,c_k}), \forall i, j, k : 0 \leq i, j < n, i \neq j, 0 \leq k < m$

for some *monotone*, *separable* and *symmetric* metric f, one can efficiently *infer* and *label* the base topology $\mathcal{G}$ induced by f over the physical topology induced through IP routing and connecting the n sources to the m clients for any sensitivity parameter $c > 0$.

As with temporally distinct observations, we have an efficient algorithm for determining whether a set of topologies are mutually consistent, and if so, generating a spanning graph G as in the definition above [2].

With these two combining mechanisms, we may merge a set of metric-topological views collected at multiple points in time and multiple points in space to produce an (unlabeled) topology incorporating the available information.

IV. Case Study: Unicast Loss Topology Identification

As established by Theorem 1, any end-to-end procedure which is capable of labeling a metric-induced topology with two clients can be recursively applied to label an arbitrarily large metric topology provided the metric is monotone and separable. However, since the exact nature of a particular procedure used to label a topology with respect to a particular metric may vary widely, the accuracy of such procedure as well as the extent to which such a labeling process scales must be analyzed in the context of the particular algorithm used. In this section, we focus our presentation on applying the theoretical results to a particular metric—namely *loss rate*. We use the term *loss topology* to refer to the instantiation of MINT for the loss rate metric.

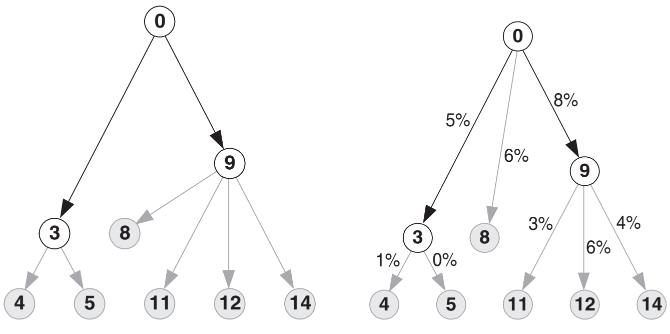
We begin this case study with a presentation of three criteria that enable us to evaluate the goodness of MINT inference and labeling. We then proceed to describe the specific procedure we used to measure shared losses between a sender and a pair of receivers. We then describe our experimental evaluation of our approach, conducted both by extensive *ns* [21] simulations and by Internet validation using Periscope—an embodiement of the MINT framework in Linux [14].

Success Criteria for Inference and Labeling: We now introduce the three criteria we use to evaluate the success of an experimental MINT inference and labeling technique. The first two evaluate the goodness of a MINT *inference* technique, whereas the third evaluates the goodness of a MINT *labeling* technique. In practice, all three of these values will be computed by running experiments over a representative set of similar trees, computing the values over those inputs, then averaging the results to derive an estimate. In each of the definitions below, we assume that the inference/labeling process starts at time $t = 0$.

Definition 6: The inference *accuracy* $A(t)$ of a MINT inference strategy at time t is the probability that the strategy yields the correct topology at time t.

In our experiments, to measure accuracy at time t, we calculate the percentage of experiments in which the correct MINT topology was identified at time t. The accuracy criterion is absolute in the sense that it does not allow for a quantification of how "close" an inferred MINT topology is to the exact MINT topology, in the event that it is inaccurate. The discrepancy measure provides such a relative quantification for tree topologies.

Definition 7: The inference *discrepancy* $D(t)$ of a MINT inference strategy on a tree topology T at time t is given by: $D(t) = \sqrt{\sum_{i,j:i\neq j}(\hat{d}_{i,j}(t) - d_{i,j})^2/\binom{M}{2}}$ where $d_{i,j}$ denotes the depth of the least common ancestor of a pair of clients i and



Inferred Tree
(a)

Labeled Tree
(b)

Fig. 2. Illustration of the use of the Inference Discrepancy and Labeling Error Measures.

j in the actual topology induced by T, $\hat{d}_{i,j}(t)$ denotes the depth of least common ancestor of a pair of clients i and j in the inferred topology at time t, and M is the number of nodes in T.

To give an intuition for the discrepancy criterion, consider the loss topology shown in Figure 1 (a) and assume that as a result of applying an inference procedure with $c = 0.05$, the topology shown in Figure 2 (left) is obtained. The inferred topology is not identical to the correct $\mathcal{T}_{0.05}$ loss topology shown in Figure 1 (d) and the *discrepancy* between the inferred topology and $\mathcal{T}_{0.05}$ is $\sqrt{3/15} = 0.447$.

Unlike quantification of the success of an inference process, which can be done independent of the metric(s) used to develop inferences, quantifying the success of a labelling process must necessarily reflect the metrical labels. The following definition applies specifically to labelled *loss* topologies.

Definition 8: The labeling *error* $E(t)$ of a loss topology labeling process on a topology T at time t is $E(t) = \sqrt{\sum_{l=1}^{L}(\hat{e}_l(t) - e_l)^2/L}$, where e_l denotes the correct loss probability (i.e. label) for link l, $\hat{e}_l(t)$ denotes the measured loss probability for link l at time t, and L is the number of links in T.

To give an intuition for the labeling error measure, which can be interpreted as the average inaccuaracy in the labeling of an arbitrary link in a topology, consider the tree shown in Figure 1 (a). Assume that the labeled topology shown in Figure 2 (right) is obtained. as a result of applying a labeling procedure with $c = 0.05$, Obviously, the labels on that topology are not identical to the labels on the $\mathcal{T}_{0.05}$ loss topology shown in Figure 1 (d). The labeling error is then $\sqrt{(0.02^2 + 0.01^2 + 0.01^2)/8} = 0.00866$.

Bayesian Probing: Our procedure for identifying shared loss between a pair of clients is the Bayesian Probing (BP) technique developed in [12]. Consider clients 11 and 14 in the topology shown in Figure 1(a). Using the terminology of Section III, paths p_{11} and p_{14} from the server to each of these clients can be partitioned into two subpaths: the portion that is *shared* between the two paths, $p_{11.14}$, and the portion that is not. Specifically, L_6L_9 is a shared segment, whereas $L_{10}L_{11}$ and $L_{13}L_{14}$ are not. The BP technique provides us with a simple probing methodology that enables the estimation of p_i, p_j and $p_{i.j}$ for all i, j as required by Theorem 1. To that end the technique uses two types of probe sequences:

Definition 9: A *1-packet probe sequence* $S_i(\Delta)$ is a sequence of packets destined to client i such that any two packets in $S_i(\Delta)$ are separated by at least Δ time units.

Definition 10: A *2-packet probe sequence* $S_{i,j}(\Delta, \epsilon)$ is a sequence of packet-pairs where one packet in each packet-pair is destined to i and the other is destined to j, and where the intra-pair packet spacing is at most ϵ and the inter-pair spacing is at least Δ time units.

1-packet probe sequences provide a baseline loss rate over end-to-end paths while 2-packet probe sequences enable measurement of loss rates over shared links. The intuition is that because of their temporal proximity, packets within a packet pair have a high probability of experiencing a shared fate on the shared links. If the incidence of shared loss on the shared links is high, this leads to an increased probability of witnessing coupled losses within a packet pair.

While we describe appropriate settings of Δ and ϵ in the experimental results, we generally find that setting ϵ to be on the order of a millisecond and Δ to be on the order of hundreds of milliseconds achieves high dependence and ensures independence, respectively. Using statistics of successful packet delivery of 1-packet and 2-packet probes, the BP techniques enables the estimation of the magnitude of packet losses on the shared segment of paths from a server to two clients.

Basic BP Assumptions: As spelled out in [12], the BP probing technique and associated analyses (leading to the estimation of packet loss rates) are subject to several significant assumptions: (1) Link loss rates are stationary over the time scale it takes the BP technique to converge. (2) Losses on the links occur only due to queue overflows. (3) $\forall i, j$: Losses on link L_i are independent from losses on link L_j. (4) There exists a reliable feedback mechanism to determine the fate of a given probe packet. (5) The temporal constraints imposed on probe sequences are preserved throughout the journey of the probes from sender to receivers. While these assumptions (with the exception of assumption 5) are satisfied in our simulation studies presented in Section IV-A, *none* could be guaranteed for our experiments over the Internet presented in Section IV-B! Nevertheless, our results show that the BP approach is robust enough to enable accurate Internet loss topology inference and labeling.

A. Performance Evaluation in ns

In this section we present results of extensive ns [21] simulations that demonstrate the accuracy, convergence and robustness of our approach.

Link Baseline Model: Each of the links in our simulations is modeled by a DropTail queue. The link delays were all set to 40ms and the link buffer sizes were all set to 20 packets. Each link was subjected to bursty background traffic resulting from aggregating a set of Pareto ON/OFF UDP sources with a constant bit rate of 36Kbps during the ON times using a packet size of 200 bytes. The average ON and OFF times were set to 2 and 1 seconds, respectively. The Pareto shape parameter (α) was set to 1.2. After a "warm-up" period of 10 seconds, the probing processes (and associated inference and labeling processes) begin.

Setting	High	Mild	Low
Link Bandwidth	1Mbps	1Mbps	100Mbps
# of background flows	60	56	44
Observed Loss Rate	7-15%	$< 7\%$	$< 1\%$

Fig. 3. Link congestion settings and resulting loss rates.

To represent the various levels of congestion that any of these links may exhibit, we have chosen three sets of parameters that reflect "High", "Mild", and "Low" levels of congestion. The baseline parameter settings for these congestion levels (and the resulting loss rates) are tabulated in Figure 3. Our choice of very high loss rates (7-15%) for highly congested links was meant to stress-test our technique under severe congestion (we observed many instances of lightly congested links in the wide area when testing our implementation). We set the value of the sensitivity parameter c to a fixed loss rate of 0.04. This value was chosen empirically based on our experimental set-up; we imagine that in general, the sensitivity parameter will have to be chosen in an application- and metric-specific way. We note that the exact upper bound on link losses is unimportant since the sensitivity parameter setting ($c = 4\%$) is small enough to observe links with losses $> 4\%$. Our simulation results are actually slightly worse with such high maximum loss rates because the presence of highly lossy links slows BP's ability to characterize subtopologies downstream of those links.

Topology Baseline Model: In order to test our inference and labeling techniques, we generated a baseline test set of random tree topologies of varying shape, depth, and with variable fanout (up to degree 4) on 14 nodes, of which 5 leaves were then selected as clients. Details of our methodology are provided in the full version of the paper [2]. The congestion level for each tree edge was then chosen at random from the link baseline models with the following distribution: 50% Low, 30% Mild and 20% High.

Configuring Bayesian Probing: The BP technique requires specification of the Δ and ϵ parameters describing the temporal constraints imposed on 1-packet and 2-packet probe sequences. In our experiments, each probe sequence was generated using an independent Poisson process with a mean probing rate of 5 probes/sec, or 200ms average inter-packet spacing. A lower bound on Δ was not guaranteed. For 2-packet probing processes, the value of ϵ was set to 0; that is packets within a packet-pair were sent back-to-back, with no time separation. The 2-packet probes in a probe sequence alternate between the two possible packet orderings.

Experimental Setup: To determine the accuracy of our loss topology inference technique, we generated 20 5-client baseline trees at random (as described earlier). We ran the loss topology inference technique from the server (root node) by creating an ns agent that sends the probes, collects statistics about these probes, calculates the needed estimates, and executes our topology inference procedure. The setup of the labeling experiment was similar; except that the ns server agent ran the labeling procedure on a provided (i.e., correct) loss topology. For each one of the 20 randomly generated trees, we ran the inference and labeling experiments 20 times; each time seeding the cross-traffic with a different random seed. The results reported below were then averaged over these 400 experiments.

Results: Figure 4 shows the accuracy and discrepancy for our loss topology inference experiments as functions of time, for different values of the sensitivity parameter c. Our inference technique converges rapidly as a function of the number of probing rounds. Figure 4 indicates that both accuracy and discrepancy settle to within 10% of their steady-state values within 50 seconds.[3]

Figure 4 (right) shows the labeling error as a function of time. The labeling error converges to within 1% of the actual links loss rates. We noted that in most of the cases the labeling results are very close to the actual losses; except for the cases where the shared links between 2 clients contain more than one highly congested bottleneck. In this case, cross-traffic packets intervene between the packet-pair probes and space the packet-pair out after the first bottleneck, causing less correlated behavior when passing through the second bottleneck. This leads to an accurate assessment of the first bottleneck loss rate while most of the second shared bottleneck loss rate is assigned incorrectly to the independent links.

Large-Scale Simulations: The above observation (regarding the effect of multiple bottlenecks) points to one of several issues which may negatively impact the performance of the BP approach to loss topology inference. Specifically, while a server can ensure that packets used in 2-packet probes are not separated by more than ϵ, it cannot "guarantee" that such a constraint is satisfied throughout the network. In practice, it is likely that the separation between packets in a packet-pair increases as the number of hops traversed increases.

To assess the robustness of our approach and its effectiveness for larger loss topologies, we have conducted experiments on a relatively large tree. We used the same tree generation procedure described in Section IV-A, except that the number of base clients was increased to 50 and the number of dummy clients was in-

[3] Since the probing rate was set to 5 probes/sec, it follows that the accuracy and discrepancy measures settle to within 10% of their steady-state values within 250 probing rounds.

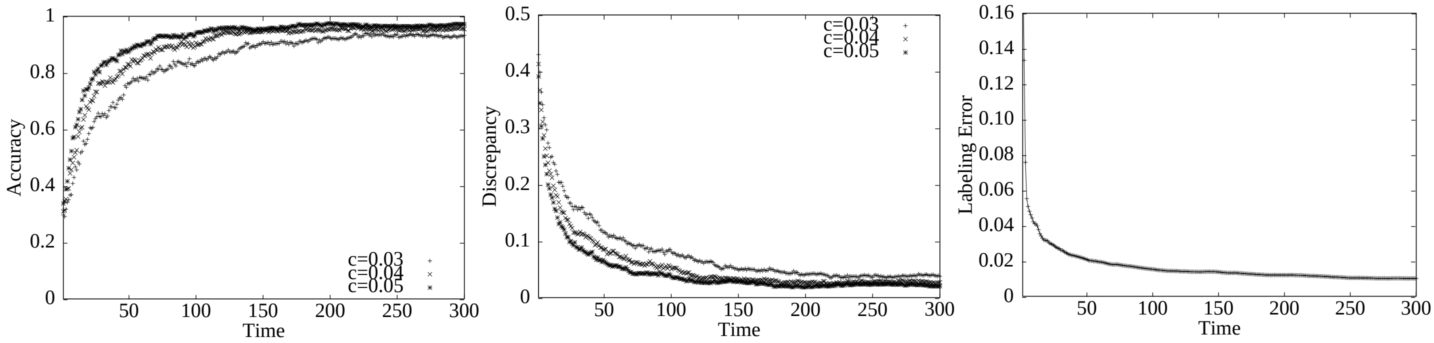


Fig. 4. Accuracy (left), Discrepancy (middle), and Labeling Error (right) of Loss Topology over Time

creased to 200. The resulting trees had an average depth of over 8 levels and in excess of 400 nodes. Also, the congestion level for each of the links of the tree was selected from one of the three link baseline models described earlier. We used a distribution with 90% Low, 7% Mild and 3% High congestion to keep the end-to-end congestion level reasonable. Results from these simulations show that while the convergence of loss topology inference for this large tree is slightly slower than that presented in Figure 4, the accuracy of the inference in steady-state remains robust (over 90% after 150 seconds). More details of these experiments are available in [2].

B. *Internet Validation using* PERISCOPE

We have embodied our MINT framework in Linux by developing an API called PERISCOPE (a Probing Engine for the Recovery of Internet Subgraphs) that implements the functionality necessary to infer and label metric-induced topologies, using the constructions presented in Section III. Details of the design and implementation of PERISCOPE as well as its use to characterize MINT topologies between a server and a set of clients across the Internet are described in [14].

To validate the ability of PERISCOPE to correctly infer and label loss topologies in an Internet setting, we hand-picked a set of seven hosts and used PERISCOPE to infer and label the loss topology to these endpoints from a local server (Pentium II processor running RedHat Linux version 2.2.14). The seven endpoints were selected to ensure the existence of different lossy paths that are shared between the server and various subsets of endpoints. In addition, by placing the server below a slow uplink, we ensured the existence of a (possibly) lossy path between the server and all endpoints. These choices were all made with the goal of stress-testing our inference and labeling techniques in mind.[4]

Figure 5 depicts the logical topology between the server and the seven hosts, constructed by collapsing chains of hops in the tree as explained in Section III-A and Figure 1). Intermediate router IP addresses were obtained through the use of *traceroute*. The server is in the continental U.S. Hosts A,B and C are in China with hosts A and B on the same LAN of Beijing University of Aeronautics and Astronautics and host C in Northeast China Institute of Electric Power Engineering. Hosts D and E are in Egypt, on the same LAN of the Arab Academy for Science and Technology (AAST). Hosts F and G are in Italy at two different universities: Politecnico di Bari and Universita Degli-Studi di Bergano.

To validate the accuracy of PERISCOPE we need to establish a "reference" against which we could compare the inferred and labeled loss trees we obtain for a given sensitivity parameter c. The logical tree (shown in Figure 5) is such a reference for $c = 0$. Obtaining such a reference for a non-zero sensitivity parameter is impossible since it requires knowledge of loss rates on all links of the logical tree. Moreover, loss rates cannot be assumed stationary for the duration of a PERISCOPE experiment and may not always be above the sensitivity parameter specified in PERISCOPE.

While the logical tree in Figure 5 cannot be used to directly validate loss trees inferred by PERISCOPE it can be used to check whether the loss trees generated by PERISCOPE are *mutually consistent*, as defined in Section III. We performed 20 experiments using PERISCOPE to infer and label the loss topology to the seven endpoints of the logical topology in Figure 5. These 20 experiments were conducted at different times. Each experiment consisted of 100 probing phases with 64-byte probes. At a probing rate of 5 probes/sec, it takes PERISCOPE about 4 minutes to complete 100 phases of probing. Notice that this time could be decreased by reducing the number of phases or by increasing the probing rate. Indeed, in most experiments, the loss topology tree "stabilized" within less than 10 phases—i.e. less than 24 seconds. However, increasing the probing rate is not desirable because it may result in the violation of the inter-probe independence assumption of the BP approach alluded to in Section IV.

Figure 6 (left) shows the percentage of PERISCOPE inferred

[4] Validating our tool requires observing loss-topologies of appreciable structure—hence our choice of an inter-continental set of endpoints. Our ongoing work on Internet loss topology characterizations to small sets of random endpoints (such as from CAIDA/NLANR logs) do not often yield rich/interesting structures. The depicted topology is meant to be illustrative, not representative. Also, experiments to the selected set of endpoints did not consistently reveal high losses. Figure 5(right) was constructed only after integrating consistent inferred loss topologies viewed at different times.

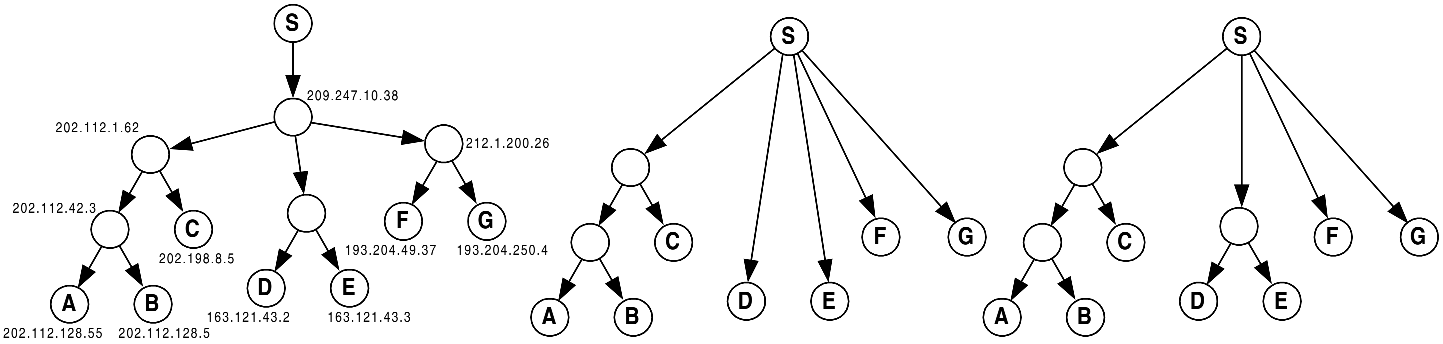


Fig. 5. PERISCOPE Validation: Logical tree used as a test case (left), most frequent inferred loss tree (middle) and minimal loss tree spanning all inferred trees (right).

trees that are found to be *inconsistent* with the logical tree in Figure 5 for various values of the sensitivity parameter c. This relationship is shown for three different periods of running PERISCOPE—namely, after 20, 40 and 80 phases. As expected, the inconsistency of the inferred tree decreases as the sensitivity parameter increases.

As explained in Section III, the non-stationarity of losses on the various links in a logical topology makes it unlikely that all of the potentially lossy links will be observable in a given experiment at a given time. Thus, one would expect that the loss topologies inferred by PERISCOPE will be different when run on the tree in Figure 5 (left). Indeed, PERISCOPE inferred six different loss topologies. Over the 20 experiments we conducted, the most frequently inferred loss topology tree is shown in Figure 5 (middle). This tree was inferred 11 times at times ranging between 3am and 7am EST (consistent with the fact that the lossy paths were the ones connecting our server to the hosts in China).

Using the procedure described in Theorem 2, we constructed the minimal loss tree spanning all six of the loss topologies inferred by PERISCOPE when $c = 0.01$. The resulting tree (which itself is not one of the trees inferred by PERISCOPE) is shown in Figure 5 (right). Clearly, that tree is consistent with the logical tree in Figure 5 (left).

To validate the labeling accuracy of PERISCOPE, we implemented a simple tool which repeatedly probes *all* nodes, including internal nodes (i.e. along subpaths to endpoints), of the logical tree of Figure 5 *concurrently* with our use of PERISCOPE. Loss statistics obtained from this Poisson probing tool are compiled to yield the loss rates on the various links of Figure 5. Finally, the resulting labeled tree is compressed using the same sensitivity parameter of PERISCOPE. The discrepancy and labeling error between the loss trees obtained using this tool and those obtained using PERISCOPE are measured and presented in Figure 6.

Figure 6 (middle) shows the discrepancy between the PERISCOPE-inferred loss trees and the loss trees obtained using the tool described above for various values of c. The results show that the discrepancy decreases slightly as c decreases.

To demonstrate the convergence characteristics of PERISCOPE, Figure 6 (right) shows the reduction in the labeling error as the number of phases executed increases from 0 to 100 phases, spanning approximately 4 minutes.

V. BANDWIDTH INDUCED NETWORK TOPOLOGIES

In Section IV, we presented an instantiation of MINT for a specific metric—namely, packet loss rate. Other instantiations of MINT are also possible and have been pursued in [11]. In this section, we briefly describe one such instantiation using the "bottleneck bandwidth" metric.

The bottleneck bandwidth (BB) of a given network path is the maximum transmission rate possible over that path (which is equal to the transmission rate of the "slowest" link along the path). The characterization of the BB topology between a set of endpoints could be quite valuable for an array of network-aware applications and services. Examples include the effective construction of ad-hoc networks in peer-to-peer communication settings, dynamic overlay networks, end-system multicast and content delivery systems.

With reference to the metric properties discussed earlier in this paper, the BB metric is *monotonic* but *not separable*. According to the results of Theorem 1, if an oracle capable of measuring the shared BB for two paths with a common endpoint exists, then it is possible to infer the BB topology between a set of endpoints. However, the complete labeling of the resulting topology is not straighforward due to the fact that the BB metric is not separable. In [13], we describe an end-to-end unicast probing technique ("Cartouche Probing") for the measurement of shared BB. Thus, using Cartouche Probing and the construction presented in Theorem 1, one can effectively infer the BB topology between a set of endpoints. Indeed Cartouche probing has been implemented in PERISCOPE, allowing for an empirical inference of BB topologies between Internet endpoints.

The absence of the separability property implies that labeling BB topologies using the (metric-independent) constructions in Theorem 1 is not possible. However, this does not imply that labeling BB topologies using other (metric-specific) constructions

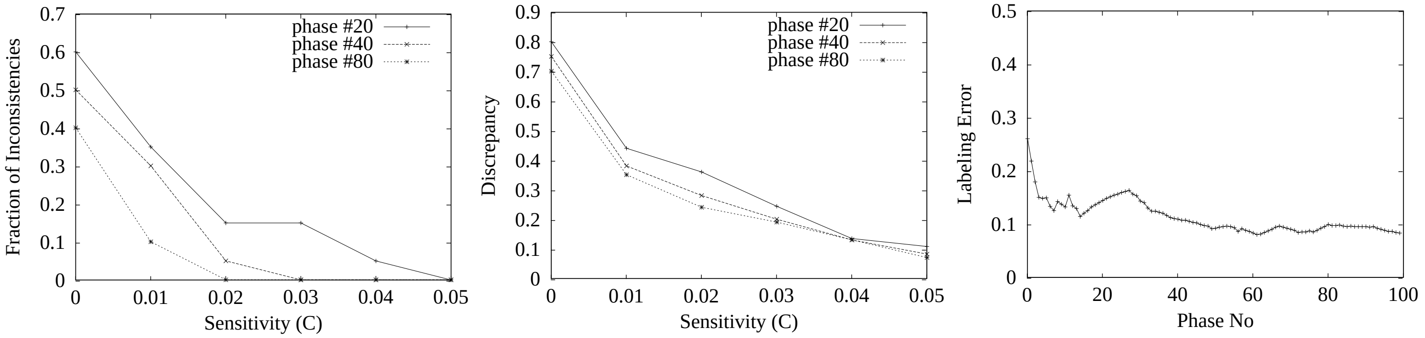


Fig. 6. Periscope: Inference inconsistency (left), discrepancy (middle) and labeling error (right) for the Internet topology of Figure 5

is not possible. Indeed, in [13] we present an efficient end-to-end approach to the measurement of BB of an arbitrary path segment, which enables a complete labeling of BB topologies, but relies on properties that are *specific* to the BB metric.

VI. Conclusion

In this paper, we presented the theoretical underpinnings of MINT—a framework for the representation of shared network resources as captured by some metrics of interest. We presented two instantiations of MINT for the packet loss rate and bottleneck bandwidth metrics. For the loss rate metric, we presented results of extensive simulations and Internet measurement experiments that confirmed the effectiveness of our constructions over a wide range of network conditions.

Our work differs from prior efforts targeted at discerning network-internal structures using end-to-end measurements (e.g., [6]) in that it is targeted at unicast applications and is applicable to classes of metrics obeying certain properties as opposed to a single metric of interest.

Our current work focuses on the use of passive unicast probing (i.e., using feedback from established TCP connections) to derive real-time MINT representations of the shared resources between a set of endpoints, and on the use of such information in making informed resource allocation decisions for network-aware applications.

References

[1] M. Allman and V. Paxson. On Estimating End-to-End Network Path Properties. In *Proceedings of ACM SIGCOMM '99*, 1999.

[2] A. Bestavros, J. Byers, and K. Harfoush. Inference and Labeling of Metric-Induced Network Topologies. Technical Report BUCS-TR-2001-010, Boston University, Computer Science Department, June 2001.

[3] J. C. Bolot. End-to-end Packet Delay and Loss Behavior in the Internet. In *SIGCOMM '93*, pages 289–298, September 1993.

[4] R. Cáceres, N. G. Duffield, J. Horowitz, D. Towsley, and T. Bu. Multicast Based Inference of Network-Internal Characteristics: Accuracy of Packet-Loss Estimation. In *INFOCOM '99*, March 1999.

[5] M. Coates and R. Nowak. Network Loss Inference Using Unicast End-to-End Measurement. In *Proceedings of ITC Conference on IP Traffic, Modelling and Management*, Monterey, CA, September 2000.

[6] N. Duffield, J. Horowitz, and F. Lo Presti. Adaptive Multicast Topology Inference. In *Proceedings of IEEE INFOCOM '01*, April 2001.

[7] N. Duffield, V. Paxson, and D. Towsley. MINC: Multicast-based Inference of Network-Internal Characteristics. http://www-net.cs.umass.edu/minc/.

[8] N. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Inferring Link Loss Using Striped Unicast Probes. In *IEEE INFOCOM 2001*, April 2001.

[9] R. Govindan and A. Reddy. An analysis of internet inter-domain routing and route stability. In *Proceedings of IEEE INFOCOM '97*, April 1997.

[10] T. Griffin and G. Wilfong. An Analysis of BGP Convergence Properties. In *ACM SIGCOMM*, pages 277–88, Cambridge, MA, September 1999.

[11] K. Harfoush. *MINT: A Framework and Toolkit for the Effective Measurement and Representation of Internet Internal Characteristics.* PhD thesis, Boston University, Computer Science Department, Boston, MA, June 2002.

[12] K. Harfoush, A. Bestavros, and J. Byers. Robust Identification of Shared Losses Using End-to-End Unicast Probes. In *8th International Conference on Network Protocols (ICNP)*, Osaka, Japan, November 2000. (Errata available as Technical Report BUCS-TR-2001-001, Boston University, Computer Science Department.).

[13] K. Harfoush, A. Bestavros, and J. Byers. Measuring Bottleneck Bandwidth of Targeted Path Segments. Technical Report BUCS-TR-2001-016, Boston University, Computer Science Department, July 2001.

[14] K. Harfoush, A. Bestavros, and J. Byers. PeriScope: An Active Measurement API. In *Proceedings of IEEE PAM'2002*, March 2002.

[15] IPMA : Internet Performance Measurement and Analysis. http://www.merit.edu/ipma.

[16] V. Jacobson. Pathchar: A Tool to Infer Characteristics of Internet Paths. ftp://ftp.ee.lbl.gov/pathchar.

[17] S. Keshav. *Congestion Control in Computer Networks*. PhD thesis, University of California at Berkeley, September 1991.

[18] K. Lai and M. Baker. Measuring Link Bandwidths Using a Deterministic Model of Packet Delay. In *SIGCOMM'00*, Aug 2000.

[19] K. Lai and M. Baker. Nettimer: A tool for Measuring Bottleneck Link Bandwidth. In *USENIX Symposium on Internet Technologies and Systems*, March 2001.

[20] Mtrace: Tracing multicast path between a source and a receiver. ftp://ftp.parc.xerox.com/pub/netsearch/ipmulti.

[21] ns: Network Simulator. http://www-mash.cs.berkeley.edu/ns/ns.html.

[22] J.-J. Pansiot and D. Grad. On Routes and Multicast Trees in the Internet. *Computer Communication Review*, 28(1):41–50, January 1998.

[23] V. Paxson. *Measurements and Analysis of End-to-end Internet Dynamics*. PhD thesis, U.C. Berkeley and Lawrence Berkeley Laboratory, 1997.

[24] S. Ratnasamy and S. McCanne. Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements. In *IEEE INFOCOM '99*, March 1999.

[25] D. Rubenstein, J. Kurose, and D. Towsley. Detecting Shared Congestion of Flows Via End-to-end Measurement. In *ACM SIGMETRICS '00*, Santa Clara, Ca, June 2000.

[26] S. Seshan, M. Stemm, and R. Katz. SPAND: Shared Passive Network Performance Discovery. In *Proc. of Usenix Symposium on Internet Technologies and Systems (USITS) '97*, Monterey, CA, December 1997.

[27] M. Yajnik, S. Moon, J. Kurose, and D. Towsley. Measurement and modelling of the temporal dependence in packet loss. In *IEEE INFOCOM '99*, pages 345–52, March 1999.