Multicast Topology Inference from End-to-end Measurements*

N.G. Duffield‡,† J. Horowitz♠ F. Lo Presti‡,§ D. Towsley§

‡AT&T Labs–Research
180 Park Avenue
Florham Park, NJ 07932, USA
{duffield,lopresti}@research.att.com

♠Dept. Math. & Statistics
University of Massachusetts
Amherst, MA 01003, USA
joeh@math.umass.edu

§Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003, USA
towsley@gaia.cs.umass.edu

Abstract

This paper describes a class of statistical estimators of multicast tree topology based on end-to-end multicast traffic measurements. This approach allows the determination of the logical multicast topology without assistance from the underlying network nodes. We provide five instances of the class, variously using loss or delay measurements. We compare their accuracy and computational cost, and recommend the best choice in each of the light and heavy traffic load regimes.

1 Introduction and Motivation

The use of multicast shows great promise for determining internal network characteristics based solely on end-to-end measurements. This is because multicast introduces correlation in the end-to-end behavior observed by different receivers within the same multicast session. This correlation can be used to estimate packet loss rates, [1], packet delay distributions, [5], and packet delay variances, [4]. These methods can be used as part of a multicast–capable measurement infrastructure, such as NIMI (National Internet Measurement Infrastructure) [9], for the purpose of monitoring internal network behavior.

All of these multicast-based methods can be applied to end-to-end multicast observations made of packets traversing a fixed, but arbitrary, topology. Knowledge of the multicast topology is required in order to apply the methods. Unfortunately, knowledge of the tree topology is not always available. This motivates the need for algorithms that can identify the topology of the multicast tree. Another motivation is that knowledge of the multicast topology can be of use to multicast applications. For example several reliable multicast protocols (e.g., RMTP [8]) rely on logical hierarchies based on the underlying topology if possible. Other applications attempt to group receivers that share the same network bottleneck, [10].

In this paper, we present a general framework within which to develop algorithms for identifying the multicast topology based on end-to-end observations. Once the topology has been identified, any of the methods mentioned above for identifying internal network behavior can then be applied. The development of an algorithm is based on the presence of a packet performance measure that monotonically increases as the packet traverses down the tree and that can be estimated based on observations made at the receivers. We provide additional constraints on the measures such that the resulting algorithm can be shown to be asymptotically consistent, i.e., it identifies the correct topology almost surely as the number of observations goes to infinity. Examples of performance measures that yield such algorithms include loss rate, delay variance, average delay, and link utilization.

Several algorithms have been proposed for identifying multicast topologies based on loss observations made at receivers. For example, [10] presented an algorithm for identifying a multicast tree when it is a binary tree. [2] established the correctness of this al-

*This work was supported in part by DARPA and the AFL under agreement F30602-98-2-0238

†Corresponding Author: Tel. +1 973 370 8726; Fax +1 973 360 8050

gorithm and introduced several other loss-based algorithms for identifying general trees. They concluded that an algorithm that constructs a binary tree and subsequently prunes branches whose loss rates are close to zero was best suited for topology identification. The framework presented in this paper comes from the recognition that this last approach can be applied to observations of other end-to-end measures such as delays.

Topology discovery is an essential part of several current measurement infrastructure projects, including CAIDA, Felix, IPMA, NIMI and Surveyor; see [3]. We contrast our approach with that of the commonly used diagnostic tools `traceroute` and `mtrace` [6] that discover physical topology. These require cooperation from intervening nodes (in the generation of ICMP messages, or in maintaining counters) and their widespread use raises issues of scaling in topologies with many leaves. The present methods complement these, being able to discover logical multicast topology and it changes without cooperation from the network, using measurement traffic whose volumes that scale well for larger topologies; see [2] for further discussion.

The paper is organized as follows. We introduce our framework in Section 2 and provide conditions under which the resulting algorithms are strongly consistent. Applications to loss and delay measures are presented in Section 3. In Section 4 we analyze the probability of topology misclassification, asymptotically for large numbers of probes. Section 5 reports on a simulation study on the effectiveness of different algorithms obtained through this approach and makes recommendations as to when they perform well. Section 6 concludes the paper.

2 A Framework for Topology Inference

Tree Model. The physical multicast tree comprises actual network elements (the nodes), and the communication links than join them. When a packet is multicast to a set of receivers, only one copy of the packet traverses each link of the tree; copies are created at the branch points of the tree, one per outgoing link. The logical multicast tree comprises the branch points of the physical tree, and the logical links between them. The logical links comprise one or more physical links. Thus each node in the logical tree, except the leaf nodes and possibly the root, must have 2 or more children.

Let $\mathcal{T} = (V, L)$ denote a logical multicast tree with nodes V and links L. We identify the root node 0 as the source of probes, and $R \subset V$ as the set of leaf nodes (identified as the set of receivers). The set of children of node $j \in V$ is denoted by $d(j)$. Each node, k, apart from the root has a parent $f(k)$ such that $(f(k), k) \in L$. Define recursively the compositions $f^n = f \circ f^{n-1}$ with $f^1 = f$. We will sometimes refer to the link $(f(k), k)$ as simply link k. Nodes are said to be siblings if they have the same parent. If $k = f^m(j)$ for some $m \in \mathbb{N}$ we say that j is descended for k (or equivalently that k is an ancestor of j) and write the corresponding partial order in V as $j \prec k$. $a(i, j)$ will denote the minimal common ancestor of i and j in the $\preceq$-ordering. For $k \in V$ we let $\mathcal{T}(k) = (V(k), L(k))$ denote the subtree of $\mathcal{T}$ that is rooted at k, and set $R(k) = R \cap V(k)$.

Tree Marks. The experience of a multicast packet on its passage down the tree is modeled by a random process of **marks** $x = (x_k)_{k \in V}$. Each mark x_k takes a value in a set $\mathcal{X}$ appropriate to the problem of interest. x_k specifies the experience of a packet traversing link k. In the setting of packet loss, for example, we take $\mathcal{X} = \{0, 1\}$, where $x_k = 1$ indicates that a packet present at node $f(k)$ is successfully transmitted to node k, while $x_k = 0$ indicates that it would be lost.

Composing Marks Along Paths. A path is a set of contiguous links, identified by the ordered set of link endpoints $(k_1, \ldots, k_\ell)$ where $k_i = f(k_{i+1})$. We will sometimes use the notation $\boldsymbol{p}(k)$ to denote the path $(k_1, \ldots, k_\ell)$ where $k_1 = 0$ and $k_\ell = k$. The experience of the multicast packet on a path is obtained by composing the marks from each link on the path to form a mark for the path. *Composition* is an associative and commutative binary operation $\otimes$ on $\mathcal{X}$. A path $\boldsymbol{p} = (k_1, \ldots, k_\ell)$ has mark $x_{\boldsymbol{p}}$ formed by successively composing the marks of its constituent links: $x_{\boldsymbol{p}} = x_{k_1} \otimes \ldots \otimes x_{k_\ell}$. We assume that $\mathcal{X}$ contains an identity $\boldsymbol{z}$ such that $\boldsymbol{z} \otimes x = x$ for all $x \in \mathcal{X}$.

In the example of loss we can compose link marks using the minimum $x_1 \otimes x_2 = x_1 \wedge x_2$. This models the physical property the loss occurs on a path if it occurs for any link on the path. The identity is $\boldsymbol{z} = 1$.

Measurements and Marks. We assume that the individual marks x_k are not directly knowable. Rather, end-to-end measurements comprise the marks $x_{\boldsymbol{p}(k)}$ along paths terminating at leaf nodes $k \in R$. Our task will be to infer the underlying topology from these path marks alone. This information can also be used to to characterize the individual links, by inferring the distribution of their link marks.

Mark Aggregation. We also equip $\mathcal{X}$ with an aggregation operation that summarizes the experience of packets over a set of possibly intersecting paths. We restrict attention to binary trees. *Aggregation* is then a binary operation $\oplus$ on $\mathcal{X}$. The multicasting property is reflected in composition $\otimes$ being distributive over aggregation $\oplus$, i.e.,

$$(x_1 \otimes x_2) \oplus (x_1 \otimes x_3) = x_1 \otimes (x_2 \oplus x_3) \qquad (1)$$

for any $x_1, x_2, x_3 \in \mathcal{X}$. The reason that this relation holds becomes clearer if we consider a four-node logical multicast tree with root 0 having a single child 1, the latter node having children $2, 3$ that are leaves. Eq. (1) says that when calculating the aggregate mark for intersecting paths $(1, 2)$ and $(1, 3)$, we can factor out the common mark on the common link 1. When dealing with loss, for example, we will take aggregation as maximization, i.e. $x_1 \oplus x_2 = x_1 \vee x_2$.

Aggregating Receiver Marks. Nodes $k \in V \setminus (R \cup \{0\})$ have two children, which we denote by $h(k)$ and $h^*(k)$. For each $k \in V \setminus \{0\}$ we define the aggregate marks over the paths to all receivers descended from k recursively through

$$\widetilde{x}_k = \begin{cases} x_{\boldsymbol{p}(k)} & k \in R \\ \widetilde{x}_{h(k)} \oplus \widetilde{x}_{h^*(k)} & \text{otherwise} \end{cases} . \qquad (2)$$

When $\oplus$ is associative (i.e. $(m_1 \oplus m_2) \oplus m_3 = m_1 \oplus (m_2 \oplus m_3)$ we can write $\widetilde{x}_k = \oplus_{j \in R(k)} x_{\boldsymbol{p}(j)}$. This is the case for loss, where $\widetilde{x}_k$ is 1 if the packet reaches some receiver descended from k and 0 otherwise. However, we have one example where $\oplus$ is not associative.

Mark Distributions. We assume that the marks are independently distributed according to a mark distribution $\omega = (\omega_k)_{k \in V}$, where ω_k is the distribution of the mark x_k. The distribution $\omega_{\boldsymbol{p}}$ of the composite mark of a path $\boldsymbol{p} = (k_1, \ldots, k_\ell)$ is determined by convolution in the usual way: for a measurable subset B of $\mathcal{X}$, $\omega_{\boldsymbol{p}}(B) = (\omega_{k_1} * \ldots * \omega_{k_\ell})(B) := \int_{x_1 \otimes \ldots \otimes x_\ell \in B} \omega_{k_1}(dx_1) \ldots \omega_{k_\ell}(dx_\ell)$. The joint distribution of $\widetilde{x}_{k_1}, \ldots, \widetilde{x}_{k_\ell}$ will be denoted $\widetilde{\omega}_{k_1, \ldots, k_\ell}$.

Deterministic Reconstruction of Binary Trees. The classification of trees relies on being able to identify certain characteristics of paths that do not terminate at leaves from the characteristics of those that do. Such a characteristic ϕ will be termed *estimable*; the precise definition is below. We seek estimable characteristics that *increase* as paths lengthen; this allows us to select as siblings those nodes for which the characteristic ϕ on the common portion of the path from 0 is maximized.

Let $\mathcal{T} = (V, L)$ be a binary tree. Let Ω be a set of probability distributions on $\mathcal{X}$ that is closed under convolution, and denote by Ω^V the corresponding product distribution of marks on $\mathcal{T}$. Let ϕ be a weakly continuous functional the set of measures on $\mathcal{X}$ that takes values in some linear space $\mathcal{Q}$. We equip $\mathcal{Q}$ with the usual partial order, i.e., $(q_i) > (q'_i)$ in $\mathcal{Q}$ iff $q_i > q'_i$ for all i. Let $\delta_{\boldsymbol{z}}$ denote the distribution which has unit mass at the composition identity $\boldsymbol{z}$.

Definition 1 *(i) ϕ is called* **estimable** *if there exists a function Φ such that, for each $\omega \in \Omega^V$, and $j, k \in V$ with $a(j, k) \neq j, k$, $\phi(\omega_{\boldsymbol{p}(a(j,k))}) = \Phi(\widetilde{\omega}_{j,k})$.*

(ii) ϕ is called **increasing** *if, $\phi(\delta_{\boldsymbol{z}}) = 0 \in \mathcal{Q}$, and for all $\omega \in \Omega$, $\phi(\omega_{\boldsymbol{p}}) < \phi(\omega_{\boldsymbol{q}})$ when $\boldsymbol{p}$ is a proper subpath of $\boldsymbol{q}$.*

The condition $\phi(\delta_{\boldsymbol{z}}) = 0$ says that a link whose marks never change the marks of paths traversing it, do not increase the value of ϕ.

Given an estimable, increasing ϕ, and a distribution ω, write $\Phi_{j,k} = \Phi(\widetilde{\omega}_{j,k})$. The topology can

```
1.  Input: The set of receivers $R = \{i_1, \ldots, i_r\}$ and
        the leaf mark distributions $\widetilde{\omega}_{i_1,\ldots,i_r}$.
2.  $R' := R$; $V' := R'$; $L' = \emptyset$ ;
3.  while $|R'| > 1$ do
4.      select $U = \{j,k\} \subseteq R'$ with maximal $\Phi_{j,k}$;
5.      $V' := V' \cup \{U\}$;
6.      $L' = L' \cup \{(U,\ell) : \ell \in U\}$;
7.      $R' := (R' \setminus U) \cup \{U\}$;
8.  enddo
9.  if $\Phi_{j,k} > 0$ do
10.     $V' := V' \cup \{0\}$ ; $L' = L' \cup \{(0,R')\}$ ;
11. enddo
12. Output: tree $(V',L')$ ;
```

Figure 1: Deterministic Binary Tree Classification Algorithm (DBT).

be reconstructed from ϕ as follows. The key observation from (ii) is that $\Phi_{j,k} > \Phi_{j',k'}$ whenever $a(j,k) \prec a(j',k')$. Thus $\Phi_{j,k}$ is maximized when j,k are siblings in R. For if not, then one of the receivers, say j, would have a sibling k' for which $\Phi_{j,k'} > \Phi_{j,k}$. Thus the siblings can be identified on the basis of leaf distributions alone. Substituting a composite node that represents their parent and iterating, should then reconstruct the binary tree. This approach is formalized in the Deterministic Binary Tree Classification Algorithm (DBT); see Figure 1.

DBT operates as follows. R' denotes the current set of nodes from which a pair of siblings will be chosen, initially equal to the receiver set R. We first find the pair $U = \{j,k\}$ that maximizes $\Phi_{j,k}$ (line 4). This identifies the members of U as siblings, and the set U is used to represent their parent. Correspondingly, we add j,k to the list V' of nodes (line 5), we add $(U,j),(U,k)$ to the list L' of links (line 6), and replace j and k by U in the set R' of nodes available for pairing in the next stage (line 7). This process is repeated until all sibling pairs have been identified (loop from line 3). Finally, we test in line 9 whether the last node grouped should be taken as the root node. If the last node identified were not the root node, equality in the test would contradict the increasing property of ϕ. Otherwise, we adjoin a root node, and a link joining it to its single descendant (line 10).

We say that DBT reconstructs the binary logical multicast tree (V,L) if given the receiver set R and the leaf mark distribution $\omega_{\boldsymbol{p}(R)}$, it produces (V,L) as its output. Clearly this happens if and only if before each iteration of the while loop 3 in Figure 1, (V',L') can be decomposed in terms of disjoint subtrees $V' = \sum_{k\in R'} V(k)$ and $L' = \sum_{k\in R'} L(k)$. These subtrees may just be trivial ones $\mathcal{T}(k) = (\{k\},\emptyset)$ comprising a root node k. We note also that these trees cover R, i.e. $R = \cup_{k\in R'} R(k)$. These properties hold before the first while loop, and hold subsequently since each loop of a successful reconstruction amalgamates binary subtrees rooted at siblings.

Theorem 1 *Let $\mathcal{T}$ be a binary tree, equipped with an estimable and increasing function ϕ. Then* DBT *reconstructs $\mathcal{T}$.*

Proof of Theorem 1: Suppose the algorithm does not reconstruct the tree. Then there must be an iteration of the while loop for which j and k in line 4 of Figure 1 are not siblings. Consider R', V' at the start of the first loop that this occurs. Let ℓ be the sibling of j. $\ell \notin R'$ since $a(j,\ell) \prec a(j,k)$ implies $\Phi_{j,\ell} > \Phi_{j,k}$, contradicting the maximality of $\Phi_{j,k}$. Since the subtrees comprising (V',L') are disjoint, no ancestor of j (or hence of ℓ) can lie in R'. Since the tree is binary, ℓ must have at least two descendents t_1, t_2 in R' since otherwise $\cup_{r\in R'} R(r)$ would not cover R. Since $a(t_1,t_2) \prec \ell$, then $\Phi_{t_1,t_2} > \phi(\omega_{\boldsymbol{p}(\ell)}) > \phi(\omega_{\boldsymbol{p}(a(j,k))}) = \Phi_{j,k}$, contradicting the maximality of $\Phi_{j,k}$. ∎

Reconstruction of Binary Trees from Measurements. Now we switch to the context that a stream of probes is dispatched from the source, each giving rise to an independent realization of the mark process. Let $x^{(i)}$ denote the i^{th} such realization. Each realization gives rise to a set of measurements $\{x^{(i)}_{\boldsymbol{p}(k)} : k \in R\}$ at the leaves. Suppose that some subtree $(V(k), L(k))$ of the tree is already identified. Then we can aggregate the measured leaf marks analogously to (2), defining $\widetilde{x}^{(i)}_k = x^{(i)}_{\boldsymbol{p}(k)}$ for $k \in R$, and forming $\widetilde{x}^{(i)}_k = \widetilde{x}^{(i)}_{h(k)} \oplus \widetilde{x}^{(i)}_{h^*(k)}$ by recursion for $k \notin R$.

Let $\widetilde{\omega}_k^{(n)} = n^{-1}\sum_{i=1}^n \delta_{\widetilde{x}_k^{(i)}}$ denote the empirical distribution of $\widetilde{x}_k^{(n)}$; here δ_y denotes the unit mass at $y \in \mathcal{X}$. We estimate $\mathcal{T}$ by the topology $\mathcal{T}^{(n)}$ obtained by using the $\widetilde{\omega}^{(n)}$ in place of $\widetilde{\omega}$ in the DBT algorithm. Specifically, we use $\Phi_{j,k}^{(n)} := \Phi(\widetilde{\omega}_{j,k}^{(n)})$ in place of $\Phi_{j,k}$ in line 3 of Figure 1. We call the resulting algorithm the Binary Tree Classification Algorithm (BT).

Theorem 2 *Under the conditions of Theorem 1, with probability* 1, $\mathcal{T}^{(n)} = \mathcal{T}$ *for sufficiently large* n. *Hence* $\mathcal{T}^{(n)}$ *is a consistent estimator of* $\mathcal{T}$ *and* $\lim_{n\to\infty} \mathsf{P}_\omega[\mathcal{T}^{(n)} \neq \mathcal{T}] = 0$.

Proof of Theorem 2: By the law of large numbers, $\omega^{(n)}$ converges weakly to ω, almost surely. Since ϕ is weakly continuous each $\Phi_{j,k}^{(n)}$ converges almost surely to $\Phi_{j,k}$. Then, almost surely, for all sufficiently large n, the relative ordering of the $\Phi_{j,k}^{(n)}$ is the same as that of the $\Phi_{j,k}$ for pairs j,k for which the $\Phi_{j,k}$ are distinct. Hence BT reconstructs the tree in the same manner as DBT, except possibly varying the order of grouping amongst sets of sibling pairs (j,k) with identical $\Phi_{j,k}$. The last two statements then follow by standard results. ∎

Characterizing Link Behavior. In many of the examples of the next section ϕ is additive over links. i.e. $\phi(\omega_1 * \omega_2) = \phi(\omega_1) + \phi(\omega_2)$. Then we can ascribe a descriptor ϕ_k, such as a packet loss rate, to each link $(f(k),k)$ through $\phi_k = \phi(\omega_{\boldsymbol{p}(k)}) - \phi(\omega_{\boldsymbol{p}(f(k))})$. (This may conveniently be done during the execution of DBT or BT).

Extension to General Trees. Inference of general trees is accomplished as follows. For simplicity assume that ϕ is additive. Then application of DBT to an arbitrary tree results in a binary tree that contains fictitious links k such that $\phi_k = 0$. The tree can then be pruned by removing any such link and identifying its endpoints. It can be shown that this procedure yields the true general tree. In BT it is necessary to apply a threshold $\varepsilon > 0$ and prune all links k with $\phi_k \leq \varepsilon$. This is because for finitely many probes, statistical fluctuations lead the characteristic ϕ of the fictitious links to differ slightly from zero. It can be shown that for sufficiently many probes, this approach reconstructs any general tree for which all $\phi_k > \varepsilon$.

3 Instances of Topology Inference

In this section we specify instances of the framework described above, specifying the setting (the marks $\mathcal{X}$, etc) and the the forms of the functions ϕ, Φ and $\Phi^{(n)}$. Theorem 1 and 2 then apply immediately in each case.

3.1 Loss-Based Inference

In this case ϕ is (a function of) the probability of successful transmission from the root to a given node. In the above formalism, we take $\mathcal{X} = \{0,1\}$, where 0 indicates packet loss and 1 transmission. Composition is by taking minima $x_1 \otimes x_2 = x_1 \wedge x_2$ and the identity is $\boldsymbol{z} = 1$; a packet is transmitted on a path if it would be transmitted on all links of that path. Aggregation is by taking maxima $x_1 \oplus x_2 = x_1 \vee x_2$; hence $\widetilde{x}_k = 1$ if a packet reaches any receiver descended from k. It can be shown [1] that

$$\mathsf{P}_\omega[x_{\boldsymbol{p}(a(j,k))} = 1] = \frac{\mathsf{P}_\omega[\widetilde{x}_j = 1]\mathsf{P}_\omega[\widetilde{x}_k = 1]}{\mathsf{P}_\omega[\widetilde{x}_j = \widetilde{x}_k = 1]}. \quad (3)$$

Using $\mathcal{Q} = \mathbb{R}_+$, we define ϕ to act on the generic measure on $\mathcal{X}$ as $\phi((1-\alpha)\delta_0 + \alpha\delta_1) = -\log(\alpha)$ for $\alpha \in [0,1]$. Clearly $\phi(\delta_{\boldsymbol{z}} = 0)$. We write $\Phi_{j,k} = \log \mathsf{P}_\omega[\widetilde{x}_j = \widetilde{x}_k = 1] - \log \mathsf{P}_\omega[\widetilde{x}_j = 1] - \log \mathsf{P}_\omega[\widetilde{x}_k = 1]$. ϕ is additive over links, and the link characteristic is $\phi_k = -\log \mathsf{P}_{\omega_k}[x_k = 1]$, i.e., the negative log probability of successful transmission over link k. Thus ϕ is increasing provided the link loss probabilities are strictly positive. For inference from measurements, $\Phi_{j,k}^{(n)} = \log n + \log(\sum_{i=1}^n \widetilde{x}_j^{(i)}\widetilde{x}_k^{(i)}) - \log(\sum_{i=1}^n \widetilde{x}_j^{(i)}) - \log(\sum_{i=1}^n \widetilde{x}_k^{(i)})$ where we have expressed the various probabilities in terms of the empirical means.

3.2 Delay Covariance-Based Inference

In this case the increasing function ϕ is the variance of the cumulative delay from the root to a given node. In the formalism, $\mathcal{X} = \mathbb{R}$, with x_k the delay encountered on link k. (The formalism extends to loss by using $x_k = \infty$ to denote loss; we treat this elsewhere [4]). Composition adds delays along a path: $x_1 \otimes x_2 = x_1 + x_2$. The identity is $\boldsymbol{z} = 0$. Aggregation takes the mean of two delays $x_1 \oplus x_2 = (x_1 + x_2)/2$. Taking $\mathcal{Q} = \mathbb{R}_+$ and $\phi(\omega) = \mathsf{Var}_\omega(x)$ we take

$$\begin{aligned}\phi(\omega_{\boldsymbol{p}(a(j,k))}) &= \mathsf{Var}_\omega(x_{\boldsymbol{p}(a(j,k))}) \\ &= \mathsf{Cov}_\omega(\widetilde{x}_j, \widetilde{x}_k) = \Phi_{j,k}.\end{aligned} \tag{4}$$

The middle equality holds since, by the independence assumption, the only non-zero contribution to $\mathsf{Cov}_\omega(\widetilde{x}_j, \widetilde{x}_k)$ is due to delays on the common portion of the paths to j and k. By the independence assumption ϕ is additive and so $\phi_k = \mathsf{Var}_\omega(x_k)$, the delay variance of link k. ϕ is increasing provided delays are not constant. $\Phi^{(n)}_{j,k}$ is the sample covariance, *i.e.*, $\Phi^{(n)}_{j,k} = \frac{1}{n}\left(\sum_{i=1}^n \widetilde{x}^{(i)}_j \widetilde{x}^{(i)}_k - \frac{1}{n}\sum_{i=1}^n \widetilde{x}^{(i)}_j \cdot \sum_{i=1}^n \widetilde{x}^{(i)}_k\right)$.

3.3 Delay Distribution-Based Inference

With inference supplying the full distribution of the cumulative delay from the root to a given node, there are several choices of the increasing function ϕ available: the complementary cumulative distribution function (ccdf), the delay moments, and the delay variance.

In the formalism, $\mathcal{X} = \{q, 2q, \ldots, dq, \infty\}$, $q > 0, d \in \mathbb{N}$, with x_k the delay on link k, discretized in bins of width q. dq is a threshold delay above which packets are considered lost, x_k taking the value ∞. Composition adds delays along a path: $x_1 \otimes x_2 = x_1 + x_2$. The identity is $\boldsymbol{z} = 0$. Aggregation takes minimum delay between paths $x_1 \oplus x_2 = x_1 \wedge x_2$; hence $\widetilde{x}_k = y$ if the minimum delay from the source to some receiver descended from k is y. In [5] it was shown how a generalization of the approach for loss inference in Section 3.1 above can be used to express the discretized distribution $\omega_{\boldsymbol{p}(a(j,k))}$ of the delay from the root to an interior node, in terms of the distribution $\widetilde{\omega}_{j,k}$ aggregate delays to leaf nodes descended through offspring j, k. More precisely, denote $A_k(i) = \mathsf{P}[x_{\boldsymbol{p}(k)} = iq]$, $\gamma_k(i) = \mathsf{P}[\widetilde{x}_{\boldsymbol{p}(k)} \le iq]$, and $\gamma_{j,k}(i) = \mathsf{P}[\widetilde{x}_{\boldsymbol{p}(j)} \oplus \widetilde{x}_{\boldsymbol{p}(k)} \le iq]$, $i \in \mathcal{X}$. Then:

$$A_{a(j,k)}(0) = \frac{\gamma_j(0)\gamma_k(0)}{\gamma_j(0) + \gamma_k(0) - \gamma_{j,k}(0)} \tag{5}$$

and $A_{a(j,k)}(i)$, $i = 1, \ldots, d$, is recursively computed as the smallest solution of the following quadratic equation:

$$\begin{aligned}&\gamma_{j,k}(i) - A_{a(j,k)}(0) + A_{a(j,k)}(0) \\ &\cdot \prod_{\ell \in \{j,k\}} \left[1 - \frac{\gamma_\ell(i) - \sum_{m=1}^{i-1} \beta_\ell(i-m) A_{a(j,k)}(m) - \beta_\ell(0) A_{a(j,k)}(i)}{A_{a(j,k)}(0)}\right] \\ &+ \sum_{m=1}^{i-1} A_{a(j,k)}(m) \left\{\prod_{\ell \in \{j,k\}} [1 - \beta_\ell(i-m)] - 1\right\} \\ &+ A_{a(j,k)}(i) \left\{\prod_{\ell \in \{j,k\}} [1 - \beta_\ell(0)] - 1\right\} = 0\end{aligned} \tag{6}$$

where $\beta_\ell(i) = \frac{\gamma_\ell(i) - \sum_{m=1}^{i} A_{a(j,k)}(m)\beta_\ell(i-m)}{A_{a(j,k)}(0)}$, $\ell \in \{j,k\}$.

In the first instance we can take $\mathcal{Q}$ as the set of ccdf's arising from the delay distributions. Excluding from Ω trivial distributions in which all link delays are zero, then since link delays are non negative, the map ϕ taking distributions to ccdf's is increasing in the sense of Definition 1(i).

In order to avoid comparing entire distributions, we can instead compare summary statistics. Since link delays are non-negative then any function of the form $\phi(\omega) = \mathsf{E}_\omega[h(x) \mid x < \infty]$ is estimable and increasing when h is an increasing function, e.g., $h(x) = x^p, p > 0$. (Here x represents a generic mark with distribution ω.) A special case is the **delay average** estimator, obtained when $p = 1$. This is additive since the mean of the sum of two random variables is the sum of their means. Another estimator is the **delay variance** estimator: $\phi(\omega) = \mathsf{Var}_\omega[x | x < \infty]$. This is additive due to the independence of link delays.

For the delay average and variance classifiers, use $\Phi_{j,k} = \mathsf{E}_\omega[x_{\boldsymbol{p}(a(j,k))} \mid x_{\boldsymbol{p}(a(j,k))} < \infty] := \sum_{i=0}^d iqA_{a(j,k)}(i) / \sum_{i=0}^d A_{a(j,k)}(i)$ and $\Phi_{j,k} = \mathsf{Var}_\omega[x_{\boldsymbol{p}(a(j,k))} \mid x_{\boldsymbol{p}(a(j,k))} < \infty] = \sum_{i=0}^d (iq)^2 A_{a(j,k)}(i) / \sum_{i=0}^d A_{a(j,k)}(i) - \mathsf{E}^2_\omega[x_{\boldsymbol{p}a(j,k)} \mid x_{\boldsymbol{p}(a(j,k))} < \infty]$, respectively, where we condition on the delay being finite. The corresponding $\Phi^{(n)}_{j,k}$ are computed using the estimated distribution $A^{(n)}_{a(j,k)}(i)$, computed through (5) and (6) using the estimates $\gamma_\ell(i)^{(n)} := n^{-1}\sum_{m=1}^n \mathbf{1}_{\{x^{(m)}_{\boldsymbol{p}(\ell)} \le iq\}}$ and $\gamma_{j,k}(i)^{(n)} := n^{-1}\sum_{m=1}^n \mathbf{1}_{\{x^{(m)}_{\boldsymbol{p}(j)} \oplus x^{(m)}_{\boldsymbol{p}(k)} \le iq\}}$ in place of $\gamma_\ell(i)$, $\ell \in \{j,k\}$, and $\gamma_{j,k}(i)$, $i \in \mathcal{X}$.

3.4 Utilization-Based Inference

In this case the increasing function ϕ is (a function of) the probability of encountering minimal delay at all links in a path. This case can be regarded as a degenerate case of the delay distribution inference in which $\mathcal{X} = \{0,1\}$ where $x_k = 0$ indicates no delay on link k, while $x_k = 1$ corresponds to any non-zero queueing delay. Hence the fraction of packets that experience $x_k = 1$ is a direct measure of the utilization of link k. Since $x_{\boldsymbol{p}(k)} = 0$ iff $x_j = 0$ for all j in the path $\boldsymbol{p}(k)$, the setup maps exactly onto the loss inference described in Section 3.1, except with the roles of 0 and 1 interchanged.

4 Misclassification Analysis

In this section, we analyze the probabilities of misclassification for the various instances of BT, and estimate their convergence rates.

Denote by E_i the event that BT has correctly reconstructed the subtree rooted at node i, $i \in V$. Since the algorithm proceeds iteratively up the tree, E_i requires first that both the subtrees rooted at its child nodes have been correctly reconstructed, then that its child nodes have been then paired together. Therefore, for $i \in V \setminus R$ we can write

$$E_i \supseteq E_{h(i)} \cap E_{h^*(i)} \cap \left(\cap_{(l,j,k)\in\mathcal{S}(i)} Q(l,j,k)\right) \quad (7)$$

where $S(i) = \{(h(i),h^*(i),k),(h^*(i),h(i),k) | i,k \neq a(i,k)\}$ and $Q(l,j,k)$ is the event that

$$D_i^{(n)}(l,j,k) := \Phi_{l,j}^{(n)} - \Phi_{l,k}^{(n)} > 0 \quad (8)$$

holds. In $\cap_{(l,j,k)\in\mathcal{S}(i)} Q(l,j,k)$, $h(i)$ and $h^*(i)$ are grouped together to form node i for all possible ways to reconstruct the tree. Denote by E the event that the tree is correctly classified. ¿From (7) we immediately have that $E \supseteq \cap_{i\in V\setminus R} \cap_{(l,j,k)\in\mathcal{S}(i)} Q(l,j,k)$. This provides the following upper bound for the misclassification probability, denoted by

$$P^f := \mathsf{P}[E^c] \leq \sum_{i\in V\setminus R} \sum_{(l,j,k)\in\mathcal{S}(i)} \mathsf{P}[Q_i^c(l,j,k)] \quad (9)$$

Normal Approximations It can be shown that $\sqrt{n}(\Phi_{i,j}^{(n)} - \Phi_{i,j})$ has asymptotically Gaussian distribution as the number of probes $n \to \infty$ in all instances described in Section 3. See [2, 4] for the loss and delay covariance case, for the other estimators it follows from Theorem 3 in [5].

Theorem 3 *Under the conditions of Theorem 1, for each $i \in V \setminus R$, $\sqrt{n} \cdot (D_i^{(n)}(j,l,k) - D_i(j,l,k))$, $(j,l,k) \in \mathcal{S}(i)$, where $D_i(j,l,k) = \Phi_{j,l} - \Phi_{j,k}$, converges in distribution, as the number of probes $n \to \infty$, to a Gaussian random variable with mean 0 and variance $\sigma_{D_i}^2(j,l,k) = \lim_{n\to\infty} n \cdot \left(\mathsf{Var}(\Phi_{j,l}^{(n)}) + \mathsf{Var}(\Phi_{j,k}^{(n)}) - 2\mathsf{Cov}(\Phi_{j,l}^{(n)}, \Phi_{j,k}^{(n)})\right)$.*

Theorem 3 suggests we can approximate $\mathsf{P}[Q_i^c(j,l,k)]$ by $\Psi\left(-\sqrt{n}\cdot\frac{D(j,l,k)}{\sigma_{D_i}(j,l,k)}\right)$, where Ψ is the cdf of the standard normal distribution. For large n, we have the following leading exponential approximation

$$\mathsf{P}[Q_i^c(j,l,k)] \approx e^{-(n/2)D_i^2(j,l,k)\sigma_{D_i}^2(j,l,k)} \quad (10)$$

where the exponent is given by the dominant term of the ratio. Since the largest term over $\cup_{i\in V\setminus R}\mathcal{S}(i)$ in (9) should dominate for large n, we expect the curve $\log P^f$ vs. n to be asymptotically linear with negative slope

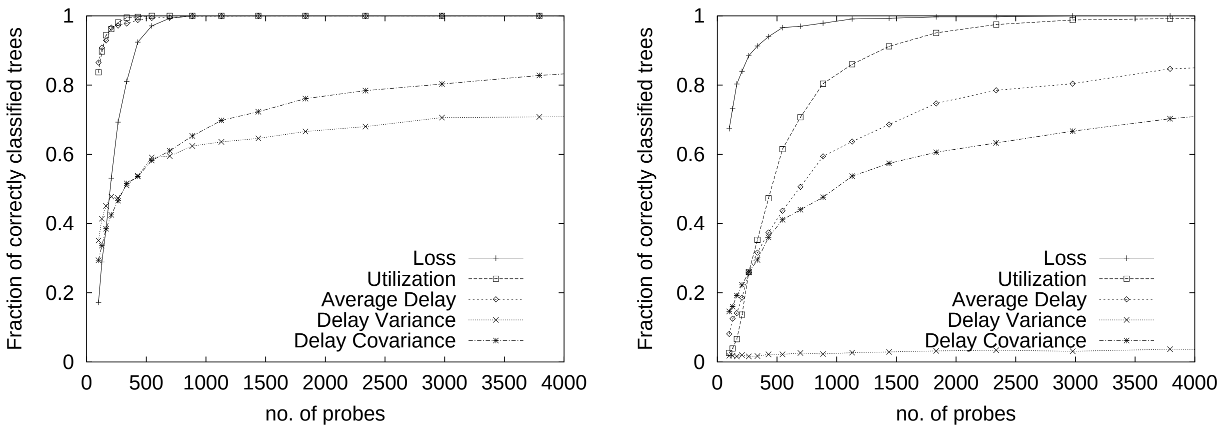
$$\inf_{i\in V\setminus R}\ \inf_{(j,l,k)\in\mathcal{S}(i)} \frac{D_i^2(j,l,k)}{\sigma_{D_i}^2(j,l,k)} \quad (11)$$

4.1 Misclassification Probability for the Different Classifiers

The foregoing analysis can be instantiated with the different estimators to obtain the misclassification probability of the topology classifiers. In the following we compute the asymptotic behavior of the different classifiers by substituting the proper expressions in (11). In general, the calculation of the infimum in (11) is quite difficult since $\sigma_{D_i}^2(j,l,k)$ is a complex function of both the topology and the distribution ω. Here, we capture the dominant modes of misclassification in asymptotic regime of small loss and delay. The results in [2] and Theorem 3 in [5] suggest that in this regime, the curve $\log P^f$ vs. n is asymptotically linear with negative slope

1. for the loss based classifier

$$\frac{n}{2}\inf_{i\in V\setminus R} \mathsf{P}[x_i = 0] \quad (12)$$



(a) (b)

Figure 2: MODEL SIMULATION. Fraction of correctly classified topologies for different classifiers as function of the number of probes: (a) light load scenario; (b) heavy load scenario.

2. for the utilization based classifier

$$\frac{n}{2} \inf_{i \in V \setminus R} \mathsf{P}[x_i > 0] \tag{13}$$

3. for the average based classifier

$$\frac{n}{2} \inf_{\substack{i \in V \setminus R \\ k \in V \mid i,k \neq a(i,k)}} \frac{\left(\sum_{i \preceq j \prec a(i,k)} \mathsf{E}(x_i)\right)^2}{\sum_{i \preceq j \prec a(i,k)} \mathsf{E}(x_i^2)} \tag{14}$$

4. for the variance based classifier

$$\frac{n}{2} \inf_{\substack{i \in V \setminus R \\ k \in V \mid i,k \neq a(i,k)}} \frac{\left(\sum_{i \preceq j \prec a(i,k)} \mathsf{Var}(x_i)\right)^2}{\sum_{i \preceq j \prec a(i,k)} \mathsf{E}(x_i^4)} \tag{15}$$

We were not able to establish an analogous result for the covariance based approach. In this case we used our experience from experiments to determine which event dominates misclassification. We observe in most experiments that, as n increases, the most likely way to misclassify a tree is by incorrectly identifying the link with the smallest link variance; this happens by mistakenly grouping one of its child nodes with its sibling node. This suggests the following approximation for the covariance approach

$$P^f \approx e^{-(n/2)\frac{\mathsf{Var}^2(x_j)}{\sigma^2_{D_j}(h(i),h^*(i),j^*)}}, \tag{16}$$

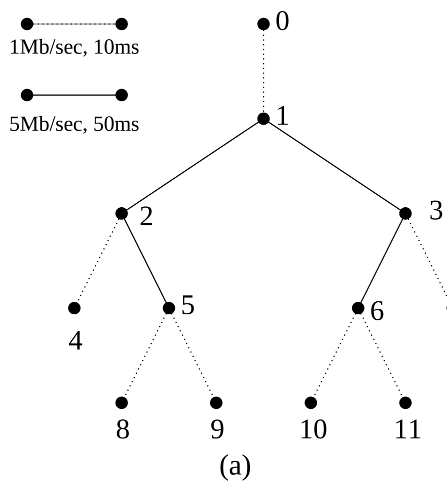
where $j = \arg\min_{i \in V \setminus R} \mathsf{Var}(x_j)$.

5 Simulation Evaluation and Algorithm Comparison

In this section we compare the performance of the different classification algorithms through two types of simulation. In *model simulations* delay and loss are chosen to follow our statistical model, allowing us to test algorithm performance in the setting on which our analysis is based. *Network simulations*, using the ns [7] simulator, test the algorithms in a more realistic setting, where delay and loss are due to queueing delay and buffer overflows at nodes as multicast probes compete with background TCP/UDP traffic.

5.1 Model Simulation

In the model simulations, at each link a probe is either lost, or encounters no delay, or suffers an exponentially distributed delay. We conducted 1000 simulations over random generated 15 node binary trees. In Figure 2 we plot the fraction of correctly classified topologies as a function of the number of probes for the different classifiers. We considered two regimes: a light load regime with low loss (1%) and utilization (randomly chosen between 10% and 40%), and a heavy load regime with higher loss (randomly chosen between 1% and 20%) and utilization (randomly chosen in between 30% and 80%). In both cases, mean delays were randomly chosen between 0.2ms



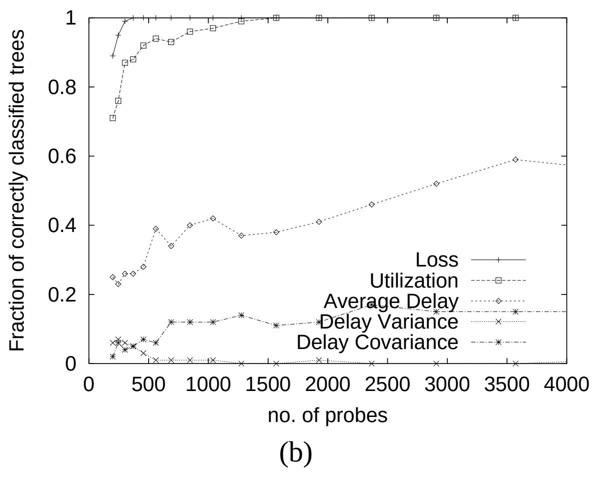


Figure 3: ns SIMULATION: (a) simulation topology; (b) fraction of correctly classified topologies for different classifiers as function of the number of probes

and 2ms. We adopted a delay granularity of 1ms. (Although this seems large compared with mean delays, the delay predictions are quite accurate; see [5]).

The loss based classifier is found to be most accurate in general. The exception is with small numbers of probes at small loss rates. Then rare losses provide insufficient data points, and accuracy is greater for the utilization and average delay classifiers. The utilization based classifier has the best accuracy among the delay distributions based classifiers. This was expected because the delay average and variance approaches use estimates of the entire delay distribution while the utilization approach uses estimates of only the first bin; estimation of the weights of lower bins being found to be more accurate. Similarly, delay average is more accurate that delay variance since it attaches less weight to higher delay bins.

From the plots, the utilization based approach appears to work very well with low links utilizations, while its performance degrades with higher utilizations, which is in contrast with (13). This can be explained by observing that the theorem holds for the limit behavior as utilization goes to 0; in our experiments, we found that (13) captures well the misclassification behavior for utilization up to 20%. On the other hand, as link utilizations increase, the number of events used by the algorithm, namely those of minimum end-to-end delay, decreases rapidly. This results in increased estimator variance.

The two best performing algorithm, (loss and utilization based) have the smallest computational complexity. All algorithms require $O(\#R^3)$ node pairs computations. Each of these is $O(n)$ for the loss, utilization and covariance based estimators. These are considerably more complex for the delay average and variance classifiers since the whole delay distribution must be calculated, by recursively solving quadratic equations, the number of which is inversely proportional to the bin size q.

5.2 TCP/UDP Network Simulation

The ns simulations used the topology shown in Figure 3(a). To capture the heterogeneity between edges and core of a WAN, interior links have higher capacity (5Mb/sec) and propagation delay (50ms) then at the edge (1Mb/sec and 10ms). Each link is modeled as a FIFO queue with a 4-packet capacity.

The root node 0 generates probes as a 20Kbit/s stream comprising 40 byte UDP packets according to a Poisson process with a mean interarrival time of 16ms; this represents 2% of the smallest link capacity. The background traffic comprises a mix of infinite data source TCP connections (FTP) and exponential on-off sources using UDP. Averaged over the different simulations, the link loss ranges between 1% and 11% and link utilization ranges between 20% and 60%. The average delay ranges between 1 and 2ms for the slower links and between 0.2 and 0.5ms for the faster links. The delay distributions were computed using a bin size of 1ms.

In Figure 3(b) we plot the fraction of correctly identified topologies over 100 simulations. The relative accuracy among the different classifiers is in agreement with the results from the model simulation with the loss based algorithm having the best performance with no misclassification for more than 500 probes. The rather poor performance of the delay based algorithms, with the exception of the utilization classifiers, is largely due to the presence of spatial correlation. In our simulations, a multicast probe is more likely to experience a similar level of congestion on consecutive links or on sibling links than is dictated by the independence assumption. This has negative impact on the accuracy of the delay estimates which accounts for the observed performance.

We also observed temporal correlation among successive probes that encountered the same congestion events. However, it can be shown that the presence of short-term correlation does not affect estimator consistency, although the convergence rate may be slowed.

6 Conclusions

In this paper we have presented a general framework for the inference of the multicast tree topologies from end-to-end measurements. In contrast with tools such as `mtrace` [6], cooperation of intervening network nodes is not required.

We specified an algorithm which reconstructs the topology of multicast tree in presence of any packet performance measure that: (i) monotonically increases as the packet traverses down the tree; and (ii) that can be estimated on the basis of end-to-end measurements at the receivers. Building on previous results in [1, 4, 5], we were able to specify several instances of this algorithm based on performance measures as packet loss, link utilization, delay average and delay variance.

We investigated the statistical properties of the algorithms, and showed that, under mild assumptions, they are consistent and computed their convergence rate. We evaluated our classifiers though simulation. We found out that the two algorithms with the lowest computational complexity, namely, the loss based and the utilization based algorithm, also have the best performance, with the loss based algorithm being in general the most accurate except when the number of probes and the loss rate are both small. Moreover, both algorithms seemed to be robust and exhibit good convergence in real traffic simulations, in spite of violation of the independence assumption of our model.

Finally, the algorithms described in this paper are each based on a different performance metric. We are currently extending our work by studying algorithms which fully take advantage of the available measurements by possibly integrating the different performance metrics we have here separately considered.

References

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

[2] R. Caceres, N.G. Duffield, J.Horowitz F. Lo Presti and D. Towsley, "Statistical Inference of Multicast Network Topology", in Proc. 1999 IEEE Conf. on Decision and Control, Phoenix, AZ, Dec. 1999.

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

[4] N.G. Duffield and F. Lo Presti, "Multicast Inference of Packet Delay Variance at Interior Network Links", Proceedings IEEE Infocom 2000, Tel Aviv, March 2000, to appear.

[5] F. Lo Presti, N.G. Duffield, J.Horowitz and D. Towsley, "Multicast-Based Inference of Network-Internal Delay Distributions", submitted for publication, September 1999.

[6] `mtrace` – Print multicast path from a source to a receiver. See `ftp://ftp.parc.xerox.com/pub/net-research/ipmulti`

[7] `ns` – Network Simulator. See `http://www-mash.cs.berkeley.edu/ns/ns.html`

[8] S. Paul, et al. "Reliable multicast transport protocol (RMTP)", *IEEE JSAC*, Vol. 15, No. 3, pp. 407–421, April 1997.

[9] V. Paxson, J. Mahdavi, A. Adams, M. Mathis, "An Architecture for Large-Scale Internet Measurement," *IEEE Communications*, Vol. 36, No. 8, pp. 48-54, August 1998.

[10] S. Ratnasamy & S. McCanne, "Inference of Multicast Routing Tree Topologies and Bottleneck Bandwidths using End-to-end Measurements", Proc. IEEE Infocom'99, New York, NY (1999)