Merging Logical Topologies Using End-to-end Measurements

Mark Coates
Department of E.C.E.
McGill University
Montreal, Quebec, Canada
coates@ece.mcgill.ca

Michael Rabbat
Department of E.C.E.
Rice University
Houston, Texas
rabbat@rice.edu

Robert Nowak
Department of E.C.E.
Rice University
Houston, Texas
nowak@rice.edu

ABSTRACT

Knowledge of network topology is useful for understanding the structure of the Internet, for developing and testing new protocols, and as prior information to network tomography algorithms. Building on existing techniques for inferring a single-source tree topology using end-to-end measurements, we address the problem of merging multiple tree topologies. We develop a multiple source active probing methodology and statistical framework for testing whether the paths from two sources to two receivers branch at a common internal node. This information can then be used to determine where portions of the tree topology from one source to a set of receivers overlap with the tree topology from a different source to the same set of receivers. The algorithm uses a novel random probing structure and easily made measurements of packet arrival order. As a result, we do not require precise time synchronization among the participating hosts. Successful experiments performed over a university LAN and over the Internet verify that our methodology is versatile and robust.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement Techniques; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*network topology*

General Terms

Algorithms, Measurement, Theory

Keywords

Network tomography, Topology discovery, End-to-end measurement, Multiple-source network tomography, Packet arrival order

IMC'03, October 27–29, 2003, Miami Beach, Florida, USA.

1. INTRODUCTION

The *physical topology* of a network describes the connectivity of the elements which comprise the network, including switches, routers, hubs and hosts. Knowledge of the physical topology of a network is extremely important for the successful execution of many network management tasks such as fault monitoring and isolation, server placement and resource sharing. The physical topology can be depicted as a graph, with internal nodes representing switching elements and edge nodes representing hosts (see the example in Figure 1). The *routing topology* of a network is related to the physical topology, and can be represented as a directed labelled graph. Over a period of routing stability the routing topology describes the paths traversed by packets sent from one end-host to another. A debate has begun within the research community as to properties of network topology graphs [13, 15]. Further understanding of these network properties will lead to improvements in the design and testing of network protocols.

Much work has been done in the area of identifying routing topologies using techniques based on information from the network BGP tables, and based on the `traceroute` program. Such work includes the Internet Mapping Project [7], the Mercator project [14], Caida's skitter project [1], and Rocketfuel [19]. These techniques hinge on eliciting special responses from internal network devices. Consequently, these techniques fail when the internal devices do not behave as expected. Internal devices such as routers, switches, and hubs may not elicit responses as expected either because they have this feature turned off (ICMP TTL Exceeded responses are optional) or because they are not capable of eliciting such responses (i.e. layer-2 devices). Barford et al. report that in experiments conducted in 2001, 13% of the internal nodes they encountered did not respond [3]. We conjecture that this number will only increase as system administrators disable this feature in routers due to rising network security concerns.

Techniques based only on end-to-end measurements avoid the problems experienced by `traceroute`-like techniques, as they do not rely on internal network devices to do anything more than route packets. However, end-to-end techniques are only able to infer a subset of the physical routing topology called the *logical topology*. Nodes in the physical topology only appear as part of the logical topology if they represent points in the network where the paths from two sources to a receiver join (*joining points*), or if they represent points where the paths from a source to two receivers

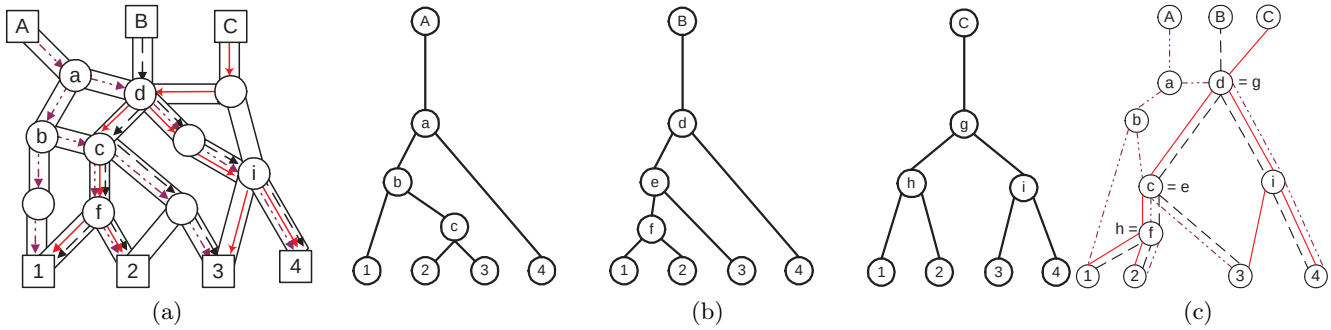


Figure 1: Physical and logical topologies of an example network. (a) The physical network showing routing paths. Circles indicate internal network elements (switches and routers), squares A-C are sources, and square 1-4 are receivers. Dot-dash lines are routes from source A, dashed lines routes from source B, and solid lines routes from source C. (b) The three logical tree topologies that can be determined from the individual sources (as might be estimated by the algorithms of [4, 8, 9]). This set of three topologies does not reflect the equivalence, or even relative position, of the nodes. In this case, node c is equivalent to node e, node d to node g, and node f to node h. The unlabelled nodes in the physical topology do not appear in the logical topologies. (c) The generalized logical topology of the multi-source network, showing the correspondence between branching nodes in the logical tree topologies. This topology clearly indicates how each source-destination path relates to all other paths.

branch (*branching points*). A single *logical link* is used to connect two such nodes if there is a (traversed) physical path between them. Logical links may encapsulate multiple physical links and nodes which are traversed consecutively. Figure 1(b) depicts logical topologies from the perspective of each source in Figure 1(a), and Figure 1(c) depicts the logical topology for the multiple source network.

While the logical topology does not describe the complete routing topology, it may still be useful for the purpose of network mapping when `traceroute`-based techniques fail. Additionally, the logical topology is relevant to *network tomography*, where end-to-end measurements are used to infer network internal properties such as delay distribution or packet drop rate. Combining topology and performance information is extremely useful for the evaluation of the resource sharing capability of the network under the current configuration, and also can guide the decisions of source-based routing algorithms. Thus, techniques which identify logical routing topologies using only end-to-end measurements are useful both for filling in the holes where other network mapping techniques fail and as an initial step in network tomography algorithms.

In this paper we build upon existing techniques which infer the logical tree topology by actively making end-to-end measurements from a single source. Specifically, we investigate the problem of merging two single-source trees from different sources to a given set of receivers in order to obtain the multiple-source, multiple receiver topology. We refer to such topologies as *general* topologies, following [6], thereby distinguishing them from the tree topologies that have been the focus of much of the logical network topology discovery literature [4, 8, 9, 11, 10, 12, 18]. The special responses elicited by techniques based on `traceroute` contain an IP address which can be used to identify internal nodes. However, because end-to-end measurements do not depend on these special responses there is no easy way to label internal nodes in an inferred logical topology such that the internal nodes inferred by one source can be related to the nodes in the logical topology inferred by another source when using end-to-end measurements. Consequently, merging two logical topologies is not a trivial task. We develop a multiple source active probing procedure and statistical framework for identifying where the paths from one source to a set of receivers enter a different source's tree topology. This information can then be used to relate internal nodes in the two trees thereby merging the single-source topologies.

The active measurement procedure we present utilizes semi-randomized probing at the sources, and packet arrival order measurements made at the receivers. As a result, no precise clock synchronization is necessary, significantly enhancing the applicability and robustness of the scheme. Based on the arrival order of packets sent from multiple hosts, the procedure makes decisions about the location where the paths from one source to the receivers join the tree topology of another source. Implementation of the algorithm is easily accomplished using either unicast or multicast packets. Additionally, because our scheme only uses end-to-end measurements, it can identify both layer-2 and layer-3 network devices. We have explored the efficacy of the algorithm through experiments in a LAN environment and over the Internet, using hosts located at universities in North America and Europe.

1.1 Related Work

A number of authors have identified techniques that rely solely on edge-based measurements to estimate the *logical* network topologies that arise when a single source communicates with multiple receivers. The papers [11, 10, 12, 18] focus on topologies reflecting the routes taken by multicast packets, whereas the papers [4, 8, 9] investigate unicast topology identification. All of the techniques assume that, from the source's point of view, the logical topology of a single-source, multiple-receiver network is a tree and is stable over the measurement period. This assumption can be violated by load balancing strategies and route changes.

The tree-oriented topology identification schemes that uti-

lize solely end-to-end measurement involve three main steps. Firstly, end-to-end measurements are made (e.g., end-to-end loss, delay, and delay differences). Secondly, a set of "end-to-end" metrics are *estimated* based on the measurements. Examples of previously used metrics include counts of joint loss events, delay covariances, and shared loss rates. In the third step of the topology identification schemes, inference algorithms use the estimated metrics to identify the topology.

A means of extending these tree identification techniques to the multiple-sender case is not clear. The schemes can obviously be used to estimate the individual tree topologies observed from each source in a multi-source tree, but the measurements do not provide enough information to enable reconstruction of the correspondence between the trees. In no technique is there a logical extension from the single-source probes to multiple-source probes that would provide additional information. In this paper, we develop a measurement framework and inference scheme that permits estimation of the connections between the single-source trees.

There are several techniques that are capable of mapping multiple-source layer-3 physical topologies, but they require that internal routers respond to ICMP requests and identify themselves using their IP addresses. The Mercator project [14], Caida's skitter project [1], and the techniques described in [3, 7] all use `traceroute` [2] in some form to determine the path from a source to a receiver. In contrast to the work presented here, these approaches focus on physical topology identification, combining traceroute measurements collected over very long time frames. A much more important distinction between these techniques and our proposed procedure is that the traceroute-based methods fail when a substantial portion of the topology is comprised of layer-2 elements (bridges and switches) or when routers do not respond to ICMP requests.

In addition to the procedures in [14, 1, 3, 7] that rely only on ICMP responses, there are other approaches that use SNMP information to generate network topology maps. Many network management tools include features that use SNMP information to map layer-3 physical topologies, e.g., IBM Tivoli Netview (www.tivoli.com). Other tools such as Cisco's Discovery Protocol (www.cisco.com) rely on vendor-specific extensions to SNMP MIB (Management Information Bases) to incorporate layer-2 elements; as a result they are applicable only in homogeneous networks (where all elements are supplied by the same vendor). Breitbart et al. [5] and Lowekamp et al. [16] describe procedures for determining physical topologies that include layer-2 elements for more heterogeneous networks. These procedures rely only on universally supported SNMP MIB information. Peregrine Systems' Infratools Network Discovery (www.peregrine.com) is a commercial tool that addresses the same task. These latter tools focus primarily on physical topology, but it is possible to derive logical topologies using them. However, all of the SNMP-based techniques require administrative access, which is typically only available to machines on the local network. The techniques can therefore only generate topology information for the component of the network where the user has administrative privileges.

2. PROBLEM STATEMENT

Two key tasks comprise the problem of identifying the unicast logical topology of a network comprised of multiple sources and multiple receivers. The first task is the discovery of the tree topologies perceived by each source. This is followed by the merger of the set of trees. Rather than developing a scheme that jointly addresses both tasks, we leverage existing techniques for identifying single-source topologies [4, 8, 9] and focus on the merging problem.

For the sake of clarity, we distill the generalized merging task into the following simpler problem and describe an approach to its solution throughout the remainder of the paper. Assume that we know (or have estimated) the logical tree topology from source C to multiple receivers. Can we determine (using end-to-end measurements) where the paths from another source A to each receiver enter the source C tree topology? This simple problem lies at the heart of the merging exercise; if we can accomplish this, then we can develop a procedure that merges multiple trees.

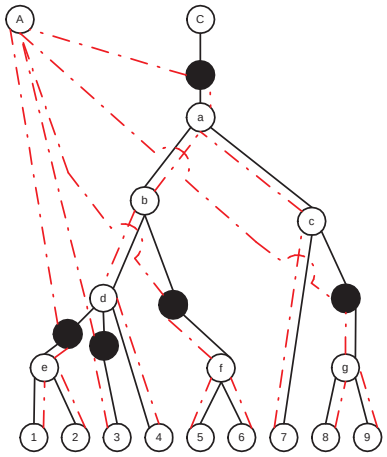


Figure 2: Nine receiver example network illustrating entry points. The solid lines and hollow circles depict the tree topology from the perspective of source C. The dashed lines and solid circles indicate where the paths from source A to the receivers join the topology (note that they do not depict the source A topology).

Figure 2 provides an illustration of the desired result, depicting a nine receiver network. The logical tree topology from the perspective of source C is shown by the solid lines and hollow circles. Our task is to identify where the paths from source A to each receiver join this tree, relative to the hollow, labelled, nodes. These entry points are shown by the solid circles. As examples, the path from A to receiver 1 enters at a point between nodes d and e, whereas the path to receiver 7 enters above node a. Observe that internal node a is the branching point for paths originating from sources A or C and going to receivers 4 and 7. We call such a node a *shared* branching point since it must be in the logical tree topologies for both sources A and C, and thus it is shared by both topologies. By knowing that a is a shared branching point, we know that the paths from A to 4 and 7 join source C's tree topology above node a. Our algorithm seeks to identify shared branching points. Two tree topologies can then be merged accordingly. Note that in this figure, the

internal nodes have been labelled only to facilitate in the problem description here and that meaningful labels do not result from any end-to-end single-source topology identification algorithm.

Our inference technique assumes: (1) *Interior switches or routers cannot be relied upon to respond to queries.* If portions of the network (for example the IP routers) do respond, then it is straightforward to incorporate the information in the discovery procedure. (2) *The topology perceived from each source is a tree.* This requires that any load balancing or routing changes over the measurement period do not affect the logical multiple-source topology. In order to make this assumption more reasonable, we seek to limit probing and keep the measurement period as short as possible. (3) *The routers and switches in the topology obey a first-in first-out policy for packets of the same class.* This is necessary to ensure that probe packets do not frequently experience reordering when traversing the same route.

2.1 Organization

In Section 3 of the paper, we describe the measurement methodology, commencing with a description and idealized analysis of a simplified two-receiver scenario. The section proceeds to conduct a more detailed analysis with more realistic assumptions, and extends the framework to multiple receiver networks. Potential extensions to the methodology are also described. Section 4 presents results from an experiment conducted on a LAN and an experiment over the Internet, two scenarios that present very different types of challenges. Section 5 discusses some limitations of the procedure and includes concluding remarks.

3. METHODOLOGY AND MEASUREMENT FRAMEWORK

3.1 A Simplified Description and Analysis

In this first description of the framework, we will perform analysis assuming no cross-traffic and clock synchronization between the sources, in order to motivate the technique and highlight the intuition behind it. In Section 3.3, we will relax these assumptions and conduct a more careful analysis with cross-traffic effects included.

We begin by exploring the simple case of a two-sender, two-receiver network. In such a network, under the assumptions outlined above, there are four possible entry scenarios, as depicted in Figure 3. Our measurement framework in this simple case proceeds as follows (in trees with more receivers, the framework is a straightforward extension). To make the n-th measurement, we send two packets from source A, spaced some small time difference Δt apart, with the first packet being sent at time t_n. The first packet, which we label $p_{A,1}$, is destined for receiver 1; the second, $p_{A,2}$, for receiver 2. We also send two packets from source C, again spaced by Δt. The first packet of this pair is sent at time $t_n + v_n$, where v_n is an offset time. The first packet, $p_{C,1}$, is sent to receiver 1 and the second, $p_{C,2}$, to receiver 2.

Figure 4(a) depicts this setup for the scenario in which the branching point is common to both sources (we will call this the *shared* scenario). Denote the fixed portion of the delay (transmission and propagation) of packet $p_{A,1}$ from source A to the joining point $d_{A,1}$, and that of packet $p_{C,1}$ from source C by $d_{C,1}$. Denote the corresponding quantities

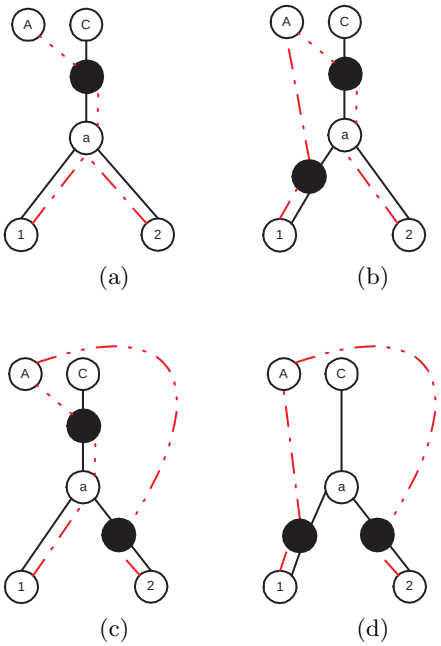


Figure 3: The four possible entry cases for a two-sender, two-receiver network. The black circles indicate entry points. Although depicted as lying in the middle of links in the C-topology, these entry points can coincide with the children nodes. For example, in (a), the entry point can be the a node, but it must lie below the C node. The dashed lines are used to indicate entry paths only, so the topology of the source A tree is not depicted except in (a). Case (a) has a common branching point for the two sources; in cases (b), (c) and (d), the branching points differ.

for the second packets sent by each source by $d_{A,2}$ and $d_{C,2}$, respectively. Since the joining point is the same in the shared scenario, $d_{A,1} = d_{A,2} = d_A$ and $d_{C,1} = d_{C,2} = d_C$. The arrival time of packet $p_{A,1}$ at the joining point is $t_n + d_{A,1}$, whereas that of packet $p_{C,1}$ is $t_n + v_n + d_{C,1}$. The arrival times of packets $p_{A,2}$ and $p_{C,2}$ are $t_n + \Delta t + d_{A,2}$ and $t_n + v_n + \Delta t + d_{C,2}$. If we now examine the arrival order of packets at the two receivers, we see that $p_{A,1}$ arrives before $p_{C,1}$ if $v_n > (d_{A,1} - d_{C,1})$. Similarly packet $p_{A,2}$ arrives before $p_{C,2}$ if $v_n > (d_{A,2} - d_{C,2})$.

We say that a measurement records a *reverse-ordering* event if the order of packet arrivals (comparing the packet from A to the packet from C) is not the same at the two receivers. In the shared branching point scenario, since $d_{A,1} = d_{A,2}$ and $d_{C,1} = d_{C,2}$, the order of arrivals at the two receivers will be exactly the same, irrespective of the offset v_n. There will be no occurrences of reverse-ordering events.

Now consider one of the *unshared* scenarios in which the branching is not common (case (b) in Figure 3). In this case, the joining points differ, so the fixed delays are (almost always) not equal, i.e., $d_{A,1} \neq d_{A,2}$ and $d_{C,1} \neq d_{C,2}$ (see Figure 4(b)). If the probes are sent at the same times as

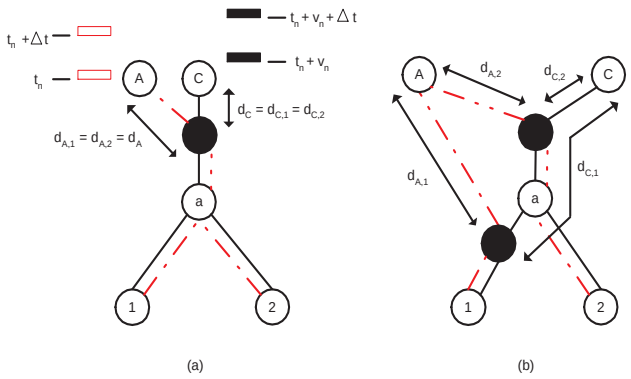


Figure 4: The measurement process. (a) Measurement for a topology in which the branching point is common. The packets next to each source are labelled with send times. The d_A and d_C labels correspond to the fixed delay component (transmission and propagation) of the indicated paths. (b) In this case the branching points are not common, the joining points differ, so the fixed delay components $d_{A,1}$ and $d_{A,2}$ are unlikely to be equal.

above, then packet $p_{A,1}$ arrives at its joining point at $t_n + d_{A,1}$, and packet $p_{C,1}$ arrives at time $t_n + v_n + d_{C,1}$. Packet $p_{A,2}$ arrives at its joining point at time $t_n + \Delta t + d_{A,2}$ and packet $p_{C,2}$ at time $t_n + v_n + \Delta t + d_{C,2}$. Let $d_1 = (d_{A,1} - d_{C,1})$ and $d_2 = (d_{A,2} - d_{C,2})$. If we compare the arrival orderings at the two receivers, we see that the orderings differ when $d_1 < v_n < d_2$ if $d_1 < d_2$, or when $d_2 < v_n < d_1$ if $d_2 < d_1$. In either case, there is an offset region of magnitude $|d_1 - d_2|$ where different orderings arise at the two receivers. The result is the same for the entry scenarios depicted in Figure 3(c) and 3(d).

The measurement process consists of repeating the measurement described above many times for $n = 1, \ldots, N$, with v_n drawn from a uniform distribution over the range $[-D, D]$ (with D chosen to be much larger than any measured round-trip-time). In the ideal world analyzed thus far, we observe no reverse-ordering events in the shared entry scenario depicted in Figure 3(a)). In the unshared scenarios of Figures 3(b)-(d), the fraction of reverse-ordering events approaches $|d_1 - d_2|/2D$ for large N. To be more precise, the number of such events obeys a binomial distribution $Bi(N, |d_1 - d_2|/2D)$.

In practice, we implement this measurement procedure by having one source send its (Δt-separated) packet-pairs at a steady rate. The rate must be sufficiently slow to avoid network flooding and probe interference. The second source sends it pairs at the same rate, but adds a random offset time (drawn from a uniform distribution over $-D, \ldots, D$). The two receivers record the orderings of packets, and send the results back to the sources.

The motivation for the spacing Δt between the two packets from each source is to ensure that they do not bunch up because of a transmission delay. If this bunching occurs, packet $p_{A,2}$ experiences additional delay relative to $p_{A,1}$ in its traversal to the joining point, so that even in the shared case $d_{A,1} \neq d_{A,2}$. Similarly, $d_{C,1} \neq d_{C,2}$. The discrepancies here are determined by the bottleneck bandwidths from the sources to the joining point; if these are not equal, then $d_1 \neq d_2$ even in the shared case, and reverse-ordering events will occur. The value Δt should thus be sufficiently large to ensure that bunching does not occur in the absence of cross traffic. We need $\Delta t > p/(\min(B_A, B_C))$, where p is the probe size, and B_A and B_C are the bottleneck bandwidths of the paths from the respective sources to the joining point. As an example, for $p = 40$ bytes and $B_A = 1$Mbps, we have $\Delta t > 320$ microseconds. In practice, we set Δt substantially larger than this to avoid as much as possible the bunching effects of cross-traffic.

The procedure just described enables us to distinguish between entry scenario (a) and entry scenarios (b)-(d) (referring to Figure 3). However, we cannot determine from these measurements exactly which of (b)-(d) is in effect. In Section 3.5, we will see that when there are more receivers in the network, it is often possible to combine the results of pairwise tests to resolve the uncertainty. We establish conditions for identifiability (localization to a single link) of the entry points.

3.2 Timing issues

The two main timing tasks involve performing an approximate synchronization of the sources at the beginning of the experiment and in keeping them on track during the experiment. Timing is not an issue at the receivers, because they simply record packet orderings.

Unless some form of synchronization is performed, the sending times of the sources will be offset from one another as a result of clock differences [17]. There will be a constant offset c_1 (in addition to the random offset v_1) between the sending times of the very first probes due to the offsets between the clocks of the two sources. In turn, the effective range of the total random offset distribution becomes $-D + c_1, \ldots, D + c_1$ rather than $-D, \ldots, D$. If we choose D such that this range still encompasses the much smaller offset region where reverse-ordering events potentially occur then the results of the experiment are unaffected by the constant offset.

If the send times are calculated naively from system clocks, then network timing protocols can induce large, unexpected shifts in relative offsets when recalibration occurs. Clock skew also arises from the physical machines having different internal system clock rates. The technique described in [17] can eliminate these problems, but as yet our procedure does not incorporate it. Over an experiment lasting a few minutes, clock drift can mean that the n-th probes are (approximately) separated by $v_n + c_1 + c_2 n$, and for the final (N-th) probe, $c_2 N$ is of the order of several hundred microseconds. The drift means that the true offset distribution is not completely uniform, but for sizeable D, it is a sufficient approximation. In fact, the use of a uniform distribution is not critical to the analysis; a distribution suffices if it satisfies the property that the ratio of the density at any two points in the range is sufficiently close to one.

Aside from the initial constant offset c_1, and the drift offset c_2 additional (and quite substantial) offsets can be incurred if the operating system swaps out the source process during an experiment. We overcome this by assigning each probe a sequence number based on the difference between the time when the experiment began and when the probe is being sent. We find that the amount of time necessary to perform some system tasks is not necessarily deterministic, but always within a small range (on the order of microsec-

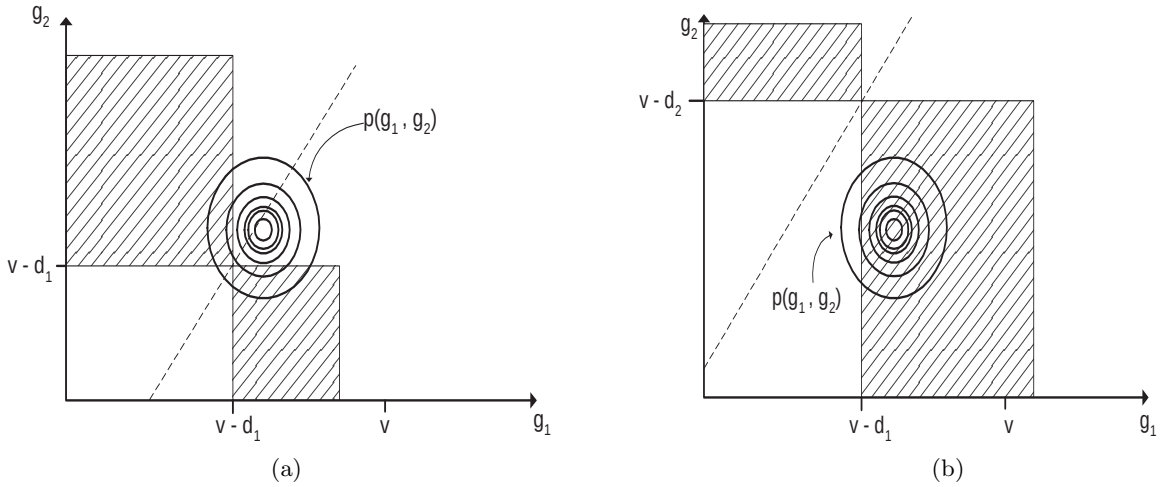


Figure 5: Cross-traffic and timing effects on ordering observations. (a) An example of how the likelihood of an ordering offset is determined for the shared scenario according to (1). The contours depict the joint probability distribution $p(g_1, g_2)$, which are the delay differences due to cross-traffic and timing error. For an offset v, the probability of a reverse-ordering event $r(v)$ is determined by the fraction of the distribution lying in the hashed regions. As v varies, the meeting point of the two subregions of integration traverses the dashed line, which passes through the origin and has slope 1. (b) The determination of the probability of a reverse-ordering event in the unshared case according to (3). In this case, as v varies, the meeting point of the subregions of integration traverses a line of slope 1 offset from the origin by $d_2 - d_1$.

onds).

While discrepancies between send-times of the first packets in corresponding pairs are evaded by choosing parameter D to be sufficiently large, it is important that the Δt values at the two sources are approximately the same. However, since Δt is only of the order of a few milliseconds, clock skew induces a maximum discrepancy of a few microseconds.

In the analysis that follows, we absorb all errors incurred by all timing discrepancies in *noise* terms that also include cross-traffic delays. An additional factor to consider in a more thorough analysis is the potential for reordering of successive probes traversing the same path can, arising, for example, as a result of multiple parallel physical connections between routers. We assume that these events are rare, because they can only occur when the packets are very closely spaced, a situation that is common in our measurement framework for only a very small range of offsets. Such reordering has the effect of very slightly increasing the probability of a reverse-ordering event.

3.3 A More Detailed Analysis

We now revisit the analysis of the arrival times for the shared scenario of the two-receiver network, incorporating cross-traffic effects. The arrival times at the joining point(s) are:

$$\begin{aligned} p_{A,1}(n) &: t_{A,1}(n) = t_n + d_{A,1} + g_{A,1}(n) \\ p_{C,1}(n) &: t_{C,1}(n) = t_n + v_n + d_{C,1} + g_{C,1}(n) \\ p_{A,2}(n) &: t_{A,2}(n) = t_n + \Delta t + d_{A,2} + g_{A,2}(n) \\ p_{C,2}(n) &: t_{C,2}(n) = t_n + \Delta t + v_n + d_{C,2} + g_{C,2}(n) \end{aligned}$$

Here $g_{A,1}(n)$ and $g_{C,1}(n)$ represent the combination of timing errors and cross-traffic delays experienced by the first packets sent from each source, and $g_{A,2}(n)$ and $g_{C,2}(n)$ are the corresponding quantities for the second packets. These terms include only the delays incurred on the path(s) to the joining point(s).

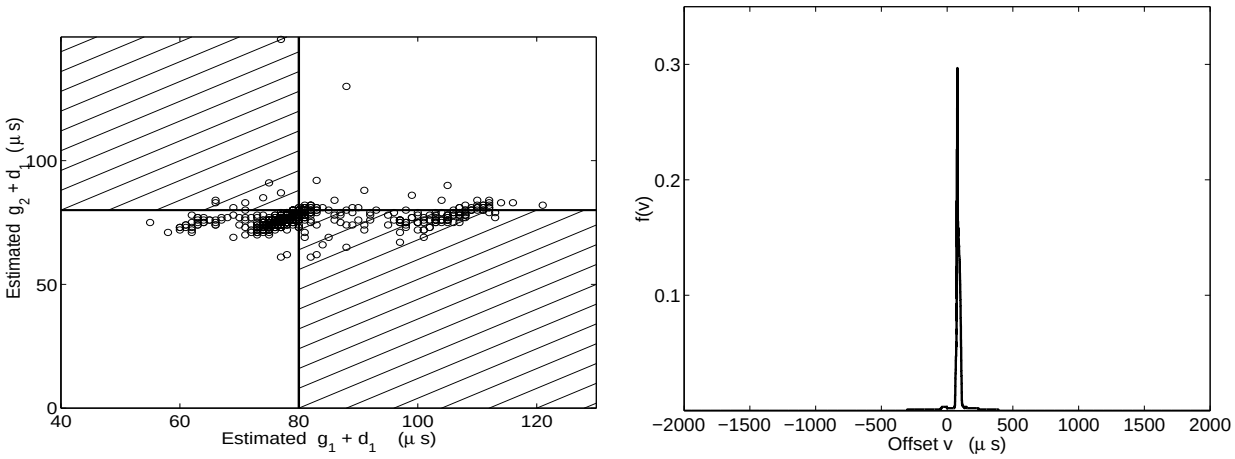
Let us first consider the shared scenario. If packet $p_{A,1}(n)$ arrives before $p_{C,1}(n)$ then $d_{A,1} + g_{A,1}(n) < v_n + d_{C,1} + g_{C,1}(n)$. Setting $d_1 = d_{A,1} - d_{C,1}$ as before, and $g_1(n) = g_{A,1}(n) - g_{C,1}(n)$, we have $v_n > d_1 + g_1(n)$. In order for a reverse-ordering event to occur, packet $p_{A,2}(n)$ must arrive after $p_{C,2}(n)$. With $d_2 = d_{A,2} - d_{C,2}$ as before, and $g_2(n) = g_{A,2}(n) - g_{C,2}(n)$, a reverse-ordering event occurs only when $v_n < d_1 + g_2(n)$, since $d_1 = d_2$ in the shared scenario. By reversing the inequalities, we obtain the expressions for the requirements for a reverse-ordering event when packet $p_{C,1}$ arrives first. If we consider a fixed offset v, the probability that a reverse-ordering event occurs is:

$$\begin{aligned} r(v) \;=\; & \int_{-\infty}^{v-d_1} \int_{v-d_1}^{\infty} p(g_1, g_2)\, dg_1\, dg_2 + \\ & \int_{v-d_1}^{\infty} \int_{-\infty}^{v-d_1} p(g_1, g_2)\, dg_1\, dg_2. \end{aligned} \tag{1}$$

The nature of this integration is depicted in Figure 5(a). At each offset point v, there is a region where an $(g1, g2)$ combination causes an reverse-ordering event. The total probability of a reverse-ordering event is then:

$$f = \frac{1}{2D} \int_{-D}^{D} r(v) dv. \tag{2}$$

Figures 6(a) and (b) display an estimation of the integral for a common branching point in the LAN experiment described in 4.1. The figure indicates the very small offset region (relative to $D = 2$ ms) where reverse-ordering can occur. The probability of a reverse-ordering event can be estimated by numerically approximating (2). For the depicted scenario, the estimated probability is 0.0017. Similar values were ob-



(a) (b)

Figure 6: (a) In the LAN experiment described in Section 4.1, delay differences were measured at a common joining point. Based on these delay differences, we estimate $g_1(n)+d_1$ and $g_2(n)+d_1$, and display them using a scatter plot. Here the hashed regions are the areas where an ordering difference would occur when the offset $v = 80 + d_1$ microseconds. (b) We estimate $f(v)$ as the fraction of points lying within the equivalent regions for each v. The estimated function $f(v)$ is displayed for $D = 2$ milliseconds. In this experiment, the estimated probability of a reverse-ordering event is 0.0017.

served for all common branching points encountered during the LAN experiment described below.

In the unshared scenarios, the arrival times at the joining points remain the same as above, but we must take into account the fact that $d_{A,1} \neq d_{A,2}$ and $d_{C,1} \neq d_{C,2}$. Proceeding as before, if packet $p_{A,1}(n)$ arrives before $p_{C,1}(n)$ then $v_n > d_1 + g_1(n)$. In order for a reverse-ordering event to occur, packet $p_{A,2}(n)$ must arrive after $p_{C,2}(n)$. This requires that $v_n < d_2 + g_2(n)$. By reversing the inequalities, we obtain the expressions for the requirements for reverse-ordering event when packet $p_{C,1}$ arrives first. If we consider a fixed offset v, the probability that a reverse-ordering event occurs is:

$$r(v) = \int_{-\infty}^{v-d_1}\int_{v-d_2}^{\infty} p(g_1,g_2)\,dg_1\,dg_2 + \int_{v-d_1}^{\infty}\int_{-\infty}^{v-d_2} p(g_1,g_2)\,dg_1\,dg_2 \quad (3)$$

The nature of this integration is depicted in Figure 5(b).

Define $P_1(t) \equiv \int_{-\infty}^{t} p_{g_1}(x)\,dx$, where p_{g_1} is the probability distribution of g_1, and equivalently, $P_2(t) \equiv \int_{-\infty}^{t} p_{g_2}(x)\,dx$. If Δt is sufficiently large, then $g_1(n)$ and $g_2(n)$ are approximately independent. In the shared case, the probability distributions are the same (assuming semi-stationarity), so $p_{g_1}(x) = p_{g_2}(x)$. Under the assumptions above we can write the following expression for f in the shared branching point scenario:

$$f(0) = \frac{1}{D}\int_t P_1(t)[1-P_1(t)]\,dt. \quad (4)$$

In the unshared scenarios, defining $d = d_1 - d_2$:

$$f(d) = \frac{1}{2D}\int_t [1-P_1(t)]P_2(t-d) + P_1(t)[1-P_2(t-d)]\,dt. \quad (5)$$

These expressions demonstrate that the probability of a different ordering event is usually much larger in the unshared case compared to the shared case. Suppose that $g_1(n)$ and $g_2(n)$ are zero-mean noises and are well concentrated (in the noise-free case they are point-mass (Dirac delta) functions located at the origin). Then $P_1(t)$ and $P_2(t)$ are approximately step functions, being near zero for $t < 0$ and close to 1 for $t \geq 0$. If this is the case and the branching point is shared, then $f(0) \approx 0$, since the integrand of (4) is zero except for a very small interval about the point $t = 0$. In the unshared case, $d \neq 0$ and $f(d) \gg 0$. To see this, note that if $0 < t < d$ (or $d < t < 0$), then the integrand of (5) is equal to 1 on a quite large interval (the size of the interval depends on the difference d). Consequently the total integral $f(d)$ is strictly greater than zero, and moreover $f(d)$ is a monotonically increasing function of d — the larger the difference d, the more distinguishable are the shared and unshared cases.

Note that the mechanism giving rise to $g_1(n)$ and $g_2(n)$ is delay variations that occur between packets which are closely spaced together. In general packet delays can vary substantially (e.g. when a burst of traffic arrives at a given queue along the path). However, for packets spaced closely together the distribution tends to be much tighter. Thus, in the shared scenario, variations in delay should rarely give rise to erroneous different arrival order events.

3.4 Setting a Threshold

After N measurements have been performed, the number of reverse-ordering events in the two receiver network test is recorded as $x_{1,2}$. Based on this value, a decision must be made as to whether the branching point is shared or not. This decision would be simpler to make if we knew how many reverse-ordering events we could reasonably expect if the branching point were shared. We can obtain an indication of this number using the following procedure. We collect measurements in exactly the same manner as the two-receiver measurement described above, except that all four

packets are sent to the same receiver. We are thus making measurements across a Y-shaped topology. We perform N measurements of this form to both receiver 1 and receiver 2 and record the number of reverse-ordering events as x_1 and x_2, respectively.

If the branching point to the receivers is shared, then the upper branches of the Y-topologies tested in these experiments coincide with the paths to the common merging point. In this case, the probability of reverse-ordering events should be the same in all three experiments, i.e., x_1, x_2 and $x_{1,2}$ are all drawn from the same binomial distribution. If the branching point is not shared, then we expect $x_{1,2}$ to be drawn from a different binomial distribution than either x_1 or x_2, and moreover, the proportion parameter of the former distribution should be significantly larger than for either of the latter distributions.

The decision as to whether a branching point is shared or unshared can now be formulated as a hypothesis test. Let $x_a = \max(x_1, x_2)$. Denote the proportion parameter of the binomial from which this measurement was drawn p_a, and the proportion parameter of the binomial from which $x_{1,2}$ was drawn $p_{1,2}$. We want to test whether these parameters are equivalent (the distributions are the same), so the hypothesis test becomes:

$$H_0: \quad p_{1,2} = p_a \tag{6}$$

$$H_1: \quad p_{1,2} > p_a \tag{7}$$

For reasonably large N, we can perform this test as a Z-test, with:

$$Z = \frac{\widehat{p_{1,2}} - \widehat{p_a}}{\sqrt{2\widehat{p}(1-\widehat{p})/N}}. \tag{8}$$

where $\widehat{p_{1,2}} = x_{1,2}/N$, $\widehat{p_a} = x_a/N$ and $\widehat{p} = (x_{1,2} + x_a)/2N$. For reasonably large N, distributions can be approximated as normal, and we can set a threshold for Z such that the probability of declaring a branching point unshared when it is in fact shared is equal to a specified level α. In our experiments, we set $\alpha = 0.05$.

3.5 Multiple Receiver Networks

Thus far, we have concentrated on describing the measurement framework for a two-receiver network. In the two receiver network, each measurement consists of a pair of packets sent from each source. The first packet from each source is destined for receiver 1, and the second for receiver 2, and there is a spacing between them of Δt. The framework for an r-receiver network is the natural extension of this. For each measurement, the two sources send a stream of r packets, with a spacing of Δt between successive packets. The i-th packet in this stream is destined for the i-th receiver. Each such measurement provides $\binom{r}{2}$ pairwise measurements of the form described above, and counts of reverse-ordering events are collected for each pair of receivers.

We perform the test described above for each pair of receivers to determine if there is only one branching point for both sources. Let $s(i,j)$ be a binary value, indicating whether receivers i and j share a common branching point from the two sources (0 indicating no, 1 indicating yes). In the simple two-receiver network, if we determined that the branching point was not shared, then it was impossible to distinguish between the three unshared entry scenarios of Figure 3(b)-(d). However, when we have multiple pairwise test results, an unshared test result can be useful information when used in conjunction with another shared test result. We apply the following simple logic algorithm to combine the results of the multiple pairwise tests.

Merging Algorithm

Step 1 The $s(i,j) = 1$ results are used to place initial bounds on the deepest points (points as close as possible to the receivers) at which the paths from A to i and j can join with the paths from C.

Step 2 Cycle through the unshared cases, $s(i,j) = 0$, and check whether or not the bounds determined in Step 1 imply more restrictive bounds on the depths for the unshared joining points. Repeat this cycle until the bounds do not change from one cycle to the next and declare convergence.

The convergence of the algorithm is guaranteed, provided that the test results do not provide conflicting evidence; see below for a discussion on how such contradictions are resolved. The proof of convergence is very simple—bounds can only be tightened, so no oscillation is possible. However, convergence of the algorithm does not mean that joining points will be localized to a single logical link. In general, the points at which the paths from source A join those of source C may only be localized to within a certain sequence of consecutive logical links in the source C tree topology.

We say that the two-source network is *identifiable* from the measurements if each point at which a path from source A joins a path from source C can be localized to a certain logical link in the source C tree topology. Conditions for identifiability are stated in the theorem below. The theorem is rather technical and slightly difficult to state, but the key point is that it demonstrates that there are many situations (conditions on the $s(i,j)$ indicator variables) in which networks are identifiable. In fact, in our experimental work described in detail in Section 4.1, the LAN we worked with was identifiable. The conditions of the theorem do not need to be checked explicitly in practice; one only needs to apply the merging algorithm above, and if the network is identifiable, then the algorithm will converge to the correct network topology.

Before stating the theorem, we introduce some necessary notation. Let $\mathcal{R}$ be the set of receivers, and let $\mathcal{D}(k)$ be the descendant receivers of node k; $\mathcal{R}/\mathcal{D}(k)$ is then the set of receivers not including $\mathcal{D}(k)$. Let C be the label of the source for which the (tree) topology is known. Let $p(k)$ be the parent of node k in this tree, and let $\mathcal{P}(i,j)$ be the path from a node i to one of its descendants j in this tree. Let $b(i,j)$ denote the branching node of the paths from C to receivers i and j. Finally, denote by b_i the first encountered branching point on $\mathcal{P}(C,i)$ for which there is a receiver $j \in \mathcal{R}$ with $b(i,j) = b_i$ and $s(i,j) = 1$. If $s(i,j) = 0$ for all $j \in \mathcal{R}/\{i\}$, then set $b_i = i$.

THEOREM 1. *A two-source network is identifiable if and only if for each receiver $i \in \mathcal{R}$ one of the two conditions holds:*

(i) $p(b_i) = C$

(ii) *there is a receiver j such that $p(b_i) = b(i,j)$ and $b_j \in \mathcal{P}(C, p(b_i))$.*

These conditions imply the requirement that there is at least one b_i with $p(b_i) = C$.

PROOF. Necessity: Suppose neither condition holds for some receiver i. Specifically, there is a receiver i such that $p(b_i) \neq C$ and that for all receivers j with $b(i,j) = p(b_i)$, $b_j \notin \mathcal{P}(C, p(b_i))$. This implies that $s(i,j) = 0$ for all such receivers j. We are now left with two possibilities for the entry point of the path to i. Either it can enter at or above $p(b_i)$, in which case the paths to each receiver j must enter below $p(b_i)$ and at or above b_j, which is possible because $b_j \notin \mathcal{P}(C, p(b_i))$. Alternatively, it can enter between $p(b_i)$ and b_i, in which case the path to receiver j can enter anywhere above b_j.

Sufficiency: If condition *(i)* holds, $p(b_i) = C$, then the path to i enters above the first branching point in the logical tree so it is localized to a single link. If not, then condition *(ii)* implies that there is a receiver j with $b(i,j) = p(b_i)$ whose path enters at or above $p(b_i)$. Furthermore, $s(i,j)$ is false (since b_i is below $b(i,j)$). This implies that the path to i cannot enter above $p(b_i)$ (otherwise $s(i,j)$ would be equal to 1). Therefore, the path enters on the link from $p(b_i)$ to b_i. In this way, each entry from source A can be localized to a single link in the tree of source C, and the network is identifiable in the sense defined above. □

A contradiction in test results will result in the algorithm attempting to make the upper bound of one of the joining points lower than the lower bound. We resolve these differences firstly by a 'majority vote' to eliminate anomalous test results. If there are equal numbers of conflicting results, then the test results are ranked by confidence (determined by Z statistics).

3.6 Extensions

The methodology and analysis presented in this paper focused on the two-source topology identification problem. Extensions to multiple source scenarios are straightforward. Beginning with a single-source tree, a second source's topological relationships are incorporated as described above. The topologies of subsequent sources can be joined to this topology, one source at a time. For each new source, the probing and merging algorithms operate in a similar manner as before, but in this case probing can be performed from the new source and any one (or all) of the other sources in the current topology. The sharedness indicators $s(i,j)$ take a non-zero value if the new source shares the i,j branching point with any one of the other sources, in which case a value indicating which source shares the branch can be assigned. The merging algorithm uses the sharedness indicators as well as their non-zero values and employs a similar cycling procedure to localize (as much as possible) the joining points for the new source.

Theorem 1 gives conditions under which the acquired measurements provide full identifiability. If these conditions are not met, then certain joining points will only be localized to within a sequence of two or more consecutive links. It may be possible to employ a more informative probing of the portion of the network in question that can help to further resolve such cases. Additional information, reflective of link bandwidths, can be gleaned by performing the procedure used to set the thresholds (Y-topology probing) but making the second packet from source C consistently much larger. When all the packets are the same size, the number of reverse-ordering events can be used to estimate $f(0)$. When one packet is much larger, however, the number of reverse-ordering events can be used to form an estimate of a metric of the path from C to the joining point. This path metric is the same as the path metric generated by the measurement procedure used in the identification of single source topologies in [9] (it is reflective of the bandwidths of the links on the path). The measurement framework in [9] can be used to determine the path metric from the source C to any branching point in the source C topology. By simply comparing the metrics of paths to branching and joining points, the relative position of all entry points can be determined. However, forming accurate estimates of the metrics can require intensive probing. For this reason, we envision that these extended measurements could form a potential secondary step, utilized only after the application of the simple and undemanding probing mechanism we have presented.

4. EXPERIMENTAL RESULTS

Our `msprobe` multiple-sender probing program implements the techniques discussed above. There are two source components and a receiver component. Source 1 sends UDP packet probes to the receivers at a regular period. Source 2 controls the experiment and sends at the same period but adds a uniform random offset to each sending time. The receiver component simply tracks the order in which probes arrive, and then sends the results back to source 2 when the experiment has reached completion. Because the only important metric is packet arrival order, no special timing infrastructure is required. After the probes have been sent the results are collected and processed at source 2. This source also keeps track of the offset used for each trial. This information can later be used, along with the outcome for each trial, to adjust the bounds of the distribution from which the offsets are chosen.

To explore the efficacy of our technique we have run experiments in two very different networking environments. The first is a departmental LAN. The second consists of hosts located at academic and research institutions throughout the United States and Europe. Each scenario presents its own set of difficulties. In the LAN, the fixed delay differences can be very small and RTTs are of the order of hundreds of microseconds, so timing issues are important and the decision-making component of the algorithm must perform well. Cross-traffic in the LAN does not produce such extreme delay variations as we observe in the Internet-wide experiment. In the Internet experiment, fixed delay differences are much larger, and RTTs of the order of tens or hundreds of milliseconds, so timing and thresholds are not so important. However, the delay variations are much larger, inducing a larger noise effect due to cross-traffic.

4.1 LAN Experiment

The first set of experiments were run over a US University departmental LAN. For this experiment there were 16 receivers with IP addresses from two different subnets. Both subnets reside over the same physical network, which consists of a single layer-3 router and multiple layer-2 ethernet switches. Figure 7 depicts the logical network connectivity of the LAN. The router is a Cisco model 6509MSFC2 and switches are 3Com SuperStack models 3300 and 1000. Note that some of the switches that interconnect hosts are

store-and-forward switches and others are cut-through. Our technique resolves shared paths regardless of the switching technology implemented at joining or branching points.

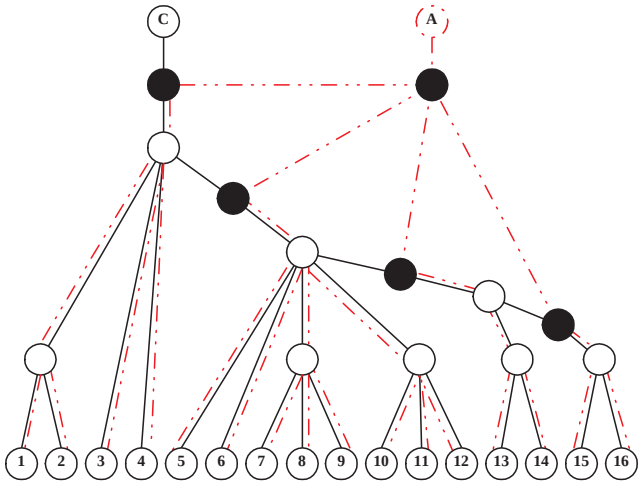


Figure 7: The true (and also discovered) logical topology of the LAN network. The hollow interior circles represent switches or routers where the paths from source C to different receivers branch apart. The filled circles indicate the nodes (the joining points) where the paths to a given receiver from sources A and C merge. In this figure, they are depicted as separate nodes, but our algorithm only resolves the location of these nodes to a single logical link of the source-C topology. If a filled node is positioned on a link in the source-C topology, then the node must lie below the parent node of that link but can either coincide with or lie above the child node.

Each probe is 68 bytes, including payload, UDP, and IP headers. We conservatively set spacing parameter Δt to be 600 microseconds based on the assumption that the minimum link bandwidth is 1Mbps. Using 600 microseconds for the random offset bound D is sufficient to encompass the range of possible delays for the short paths of the LAN.

In our experiments on this topology, all of the decisions (shared or unshared branching points) were correct in the sense that they agreed with the known logical connectivity. The decisions were made using the methodology for setting thresholds described in Section 3.4. Figure 8 graphically depicts the results of one experiment. We correctly identify the set of shared paths. In this case, the results are sufficient to completely resolve (to the logical link level) where the paths from source A to the receivers join those from source C.

4.2 Internet Experiment

In order to explore algorithm performance in an environment very different from the LAN, we performed another set of experiments using Internet hosts located in North America and Europe. For these experiments there were 9 receiving hosts located at 5 different academic establishments. The two sources were both situated in North America. Figure 9 shows the logical connectivity between sources and receivers, identified using the `traceroute` program.

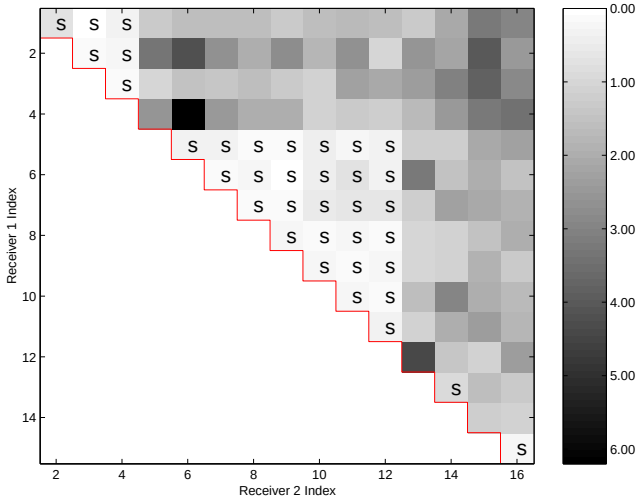


Figure 8: Results of the LAN experiment. The x- and y-axes correspond to receivers as labelled in Figure 7. The shade of gray of the square at position (i, j) indicates the observed ratio of different ordering events to total measurements for the receiver pair (i, j). If the square at (i, j) is labelled with an "s", then the paths from the two sources to receivers i and j share a common branching point in the true topology. When the detection threshold is set to 1.00, the value determined by the procedure outlined in Section 3.4, then all test decisions are correct.

The major network properties that affect parameter selection for our technique are minimum link bandwidth and maximum end-to-end delay. Because these properties differ greatly between the LAN and Internet scenarios, software parameters need to be adjusted accordingly. The same 68 byte UDP probes are used in either case. To account for a potentially lower minimum link bandwidth we increase the packet spacing parameter Δt from 600 microseconds to 1 millisecond. Likewise, to adjust for the much larger range of possible end-to-end delays the random offset is drawn from a uniform distribution spanning 90 milliseconds.

In this experiment we are able to correctly identify pairs of receivers with shared paths from the two sources, but not completely resolve entry points. Figure 10 shows the results. In the Internet experiments, the set of results is insufficient to resolve the entry points of the paths from source A to a single link. More receivers are required to produce a more complete picture.

5. DISCUSSION AND CONCLUSIONS

We have presented a technique for identifying shared paths from multiple senders to a receiver using only end-to-end measurements. This information can then be used to merge two single-source tree topologies. The framework we propose revolves around a randomized probing scheme, with receivers only recording packet arrival order. Without the need for precise timing measurements, our scheme is very practical to implement. Through Internet and LAN experiments we have demonstrated the versatility and robustness of the technique.

The experiments we report involve a relatively small num-

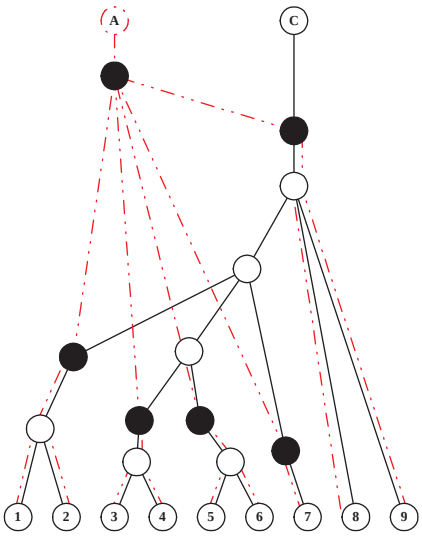


Figure 9: True logical topology of the Internet experiment testbed. Shared branching points only occurred when both receivers were physically located on the same campus, i.e. receiver pairs (1,2), (3,4), (5,6) and (8,9). In this case the network topology is not identifiable in the sense we defined above. In this experiment, we cannot completely resolve the entry points of the paths from A, but we do correctly identify shared branching points.

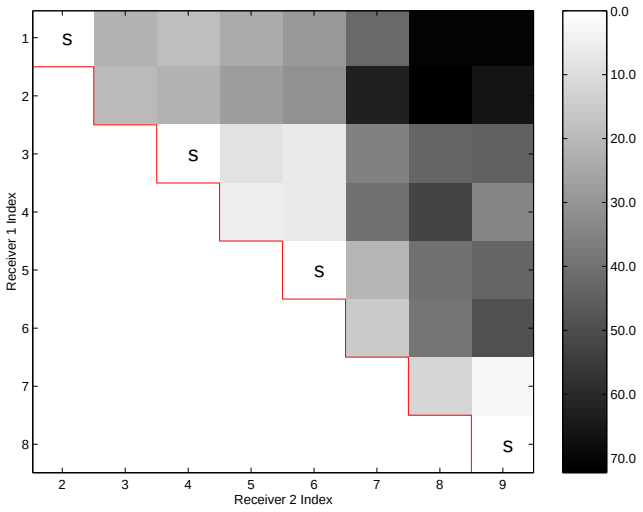


Figure 10: Results of an Internet experiment. Note that in comparison to the LAN experiment, the ratio reverse-orderings spans a much greater range. This can be attributed to two factors: (1) the fixed delay differences d_1 and d_2 are much larger in the Internet and (2) the range of end-to-end delays experienced by packets on the Internet is much larger than in a LAN.

ber of receiver hosts. Admittedly, techniques using only end-to-end measurements do not scale well to large numbers of receivers. For a network consisting of M sources and N receivers, `traceroute`-based techniques require $\mathcal{O}(MN)$ measurements to be made (one for each source-receiver pair). Using end-to-end multicast measurements, our technique requires $\mathcal{O}\left(\binom{M}{2}N\right)$ measurements. Thus, there is a tradeoff between relying on special purpose responses from internal network elements and using end-to-end techniques which require more measurements. However, in situations where the network does not facilitate the use of `traceroute`-based techniques, an algorithm using end-to-end measurements to infer the logical topology may be better than nothing at all. Additionally, while it may not be practical to only make measurements to pairs of receivers at a time for large numbers of receivers, we believe this work offers an important incite as to how algorithms based on end-to-end measurements, such as our multiple source algorithm, can potentially be used to fill in where other measurement methodologies leave off.

In future work, we will explore the development of multiple source probing methods aimed at characterizing network topology and performance. We also plan to investigate the extent to which measurements made from multiple sources can be used to infer topology without knowledge of any single-source tree topologies.

6. REFERENCES

[1] Skitter. http://www.caida.org/tools/measurement/skitter.

[2] traceroute. http://www.traceroute.org.

[3] P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *Proc. IEEE/ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, Nov. 2001.

[4] A. Bestavros, J. Byers, and K. Harfoush. Inference and labeling of metric-induced network topologies. Technical Report BUCS-2001-010, Computer Science Department, Boston University, June 2001.

[5] Y. Breitbart, M. Garofalakis, C. Martin, R. Rastogi, S. Seshadri, and A. Silberschatz. Topology discovery in heterogeneous ip networks. In *Proc. IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000.

[6] T. Bu, N. Duffield, F. L. Presti, and D. Towsley. Network tomography on general topologies. In *Proc. ACM Sigmetrics*, Marina Del Rey, CA, Jun. 2002.

[7] H. Burch and B. Cheswick. Mapping the Internet. *IEEE Computer*, 32(4):97–98, 1999.

[8] R. Castro, M. Coates, and R. Nowak. Maximum likelihood identification of network topology from end-to-end measurements. In *DIMACS Workshop on Internet and WWW Measurement, Mapping and Modeling*, Piscataway, NJ, Feb. 2002. Extended version available as Rice University ECE Tech. Rep. TR-0109, www.spin.rice.edu/publications.html.

[9] M. Coates, R. Castro, M. Gadhiok, R. King, Y. Tsang, and R. Nowak. Maximum likelihood network topology identification from edge-based unicast measurements. In *Proc. ACM Sigmetrics*, Marina Del Rey, CA, Jun. 2002.

[10] N. Duffield, J. Horowitz, and F. L. Presti. Adaptive multicast topology inference. In *Proceedings of IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[11] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Multicast topology inference from end-to-end measurements. In *ITC Seminar on IP Traffic, Measurement and Modelling*, Monterey, CA, Sep. 2000.

[12] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Multicast topology inference from measured end-to-end loss. *IEEE Trans. Info. Theory*, 48(1):26–45, January 2002.

[13] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, 1999.

[14] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *Proc. IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000.

[15] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling bases in ip topology measurements. In *Proceedings of IEEE Infocom 2003*, San Francisco, CA, April 2003.

[16] B. Lowekamp, D. O'Hallaron, and T. Gross. Topology discovery for large ethernet networks. In *Proc. ACM SIGCOMM 2001*, San Diego, CA, Aug. 2001.

[17] A. Pásztor and D. Veitch. A precision infrastructure for active probing. In *Proc. Workshop on Passive and Active Newtorking*, Amsterdam, Apr. 2001.

[18] S. Ratnasamy and S. McCanne. Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements. In *Proceedings of IEEE INFOCOM 1999*, New York, NY, March 1999.

[19] N. Spring, R. Mahajan, and D. Wetherall. Measuring isp topologies with rocketfuel. In *Proceedings of ACM/SIGCOMM '02*, Aug. 2002.