Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 1994–2009
computer communications
www.elsevier.com/locate/comcom

# Topology-aware overlay path probing

Chiping Tang [a], Philip K. McKinley [b,*]

[a] *Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, USA*
[b] *Department of Computer Science, Michigan State University, East Lansing, MI 48824, USA*

Received 6 June 2006; received in revised form 8 March 2007; accepted 8 March 2007
Available online 13 March 2007

**Abstract**

Path probing is essential to maintaining an efficient overlay network topology. However, the cost of a full-scale probing is as high as O($n^2$), which is prohibitive in large-scale overlay networks. Several methods have been proposed to reduce probing overhead, although at a cost in terms of probing completeness. In this paper, an orthogonal solution is proposed that trades probing overhead for estimation accuracy in sparse networks such as the Internet. The proposed solution uses network-level path composition information (for example, as provided by a topology server) to infer path quality without full-scale probing. The inference metrics include latency, loss rate and available bandwidth. This approach is used to design several probing algorithms, which are evaluated through extensive simulation. The results show that the proposed method can reduce probing overhead significantly while providing *bounded* quality estimations for all of the $n \times (n-1)$ overlay paths. The solution is well suited to medium-scale overlay networks in the Internet. In other environments, it can be combined with extant probing algorithms to further improve performance.
© 2007 Elsevier B.V. All rights reserved.

*Keywords:* Overlay network; Network measurement; Link sharing; Quality inference; Path segment; Set cover

## 1. Introduction

Overlay networks have recently attracted attention, due to their ability to support applications such as end-system multicast [1,2], structured peer-to-peer systems [3,4], global event notification [5], resilient routing [6], and DOS (denial-of-service) attack prevention [7]. Overlay applications exploit the high flexibility, deployability, and computing capability of end systems to improve network services.

The choice of overlay network topology has significant effect on system performance. For example, a suboptimal multicast tree may result in high link stress (number of packets crossing a link) and high RDP (relative delay penalty) [1], while a poor-quality backup path provides little fault tolerance. In order to construct a good overlay topology in dynamic environments such as the Internet, overlay nodes need to periodically *probe* the paths to other nodes and monitor the qualities of those paths. A wide variety of probing techniques and tools are available to measure latency, loss rate, and bandwidth on individual paths [8]. The overlay network can use these methods to determine which of the $n \times (n-1)$ paths should be included in the overlay topology.

One straightforward solution is pair-wise probing, which is used in Narada [1] and RON [6]. Although this approach is complete and accurate, the number of probes, and therefore the probing overhead, is O($n^2$), where $n$ is the number of overlay nodes. In a sparse network such as the Internet [9], overlay paths are likely to share physical links, and thus pair-wise probing may produce high link stress. The worst-case link stress is O($n^2$), where it happens that a single "bridge" link connects two subnetworks that equally partition the overlay nodes. Although this is an extreme case, our simulation results show the expected worst-case link stress can still be considerably higher than O($n$) in real networks. For example, Fig. 1 plots link stress for pair-wise overlay path probing in an AS-level Internet topology, as6474 [10] (note both axes are logarithmic;

[*] Corresponding author. Tel.: +1 517 353 4396.
*E-mail address:* mckinley@cse.msu.edu (P.K. McKinley).

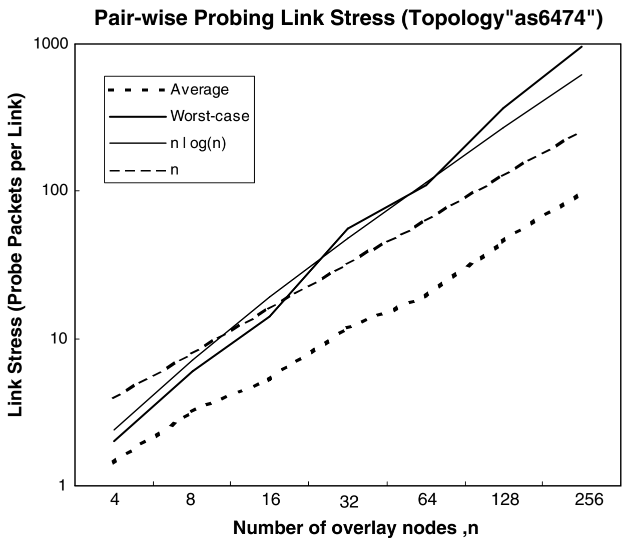


Fig. 1. Link stress of pair-wise probing, Internet AS topology, February 2000.

topology details are provided in Section 5). High link stress not only affects the normal data traffic on that link, but also affects the probing results due to link sharing among overlay paths.

In this paper, we propose a probing strategy that can significantly reduce probing overhead from O($n^2$) while providing a *bounded* quality estimation for all of the $n \times (n-1)$ paths. The method, *MPROBE*[1], uses network topology information and exploits link sharing among overlay paths. The general approach is based on the following observations: (1) the latency of a path is greater than or equal to the latency of any of its subpaths; (2) the loss rate of a path is greater than or equal to the loss rate of any of its subpaths; and (3) the available bandwidth of a path is lower than or equal to the available, or bottleneck, bandwidth of any of its subpaths. After probing a path, the method computes coarse quality estimates for all physical links on that path, from which can be inferred the quality of other paths containing those links.

The main focus of this paper is on the foundations of MPROBE, including description of the inference techniques, presentation of specific algorithms, and evaluation of the proposed methods through simulations. Ongoing studies demonstrate the effectiveness of these methods when used to support communication services on the PlanetLab Internet testbed [11]. MPROBE is best suited to overlay applications that require bounded, rather than precise, estimations for all paths. A typical example is RON [6], in which each node periodically probes all other nodes to find possible alternative paths. Instead, MPROBE can provide a bounded quality estimation for every path, while incurring much lower probing overhead. If the goal of path probing is simply to bypass path outages, then the estimation bound can be very loose, since any path with finite latency will be acceptable.

The performance of MPROBE depends on the following assumptions: (1) the degree of link sharing between overlay paths is relatively high; (2) route changes are less frequent than path quality changes; and (3) the network-level composition of every overlay path is known by at least one overlay node. Assumption (1) is self-evident in the Internet, where the average node degree is O(1) [9]. Assumption (2) is generally true, since a route change is usually caused by path quality change, while the reverse does not necessarily hold. Although assumption (3) has not generally held in the past, end node techniques such as *traceroute* and, more recently, *topology servers* [12], can be used to obtain network topology information. In addition, this information can be inferred from end-to-end measurements [13,14]. Indeed, several recent studies assume the availability of topology information at end nodes [15,16]. Moreover, we shall demonstrate that MPROBE works with gradually degraded performance even when the topology information is incomplete.

The main contributions of this work are threefold. First, we propose a practical approach to solving the overlay path probing problem. Compared with existing methods, MPROBE can increase probing completeness, while requiring comparable or lower probing overhead, and provide reasonable estimation accuracy. Moreover, MPROBE is orthogonal to existing approaches and thus can be combined with them to further improve performance. Second, we introduce a suite of probing algorithms that trade probing cost for estimation accuracy. Third, we evaluate the strengths and weaknesses of these algorithms, and of MPROBE in general, through simulation. Both real and generated Internet topologies are studied, combined with different network conditions. Metrics of interest include latency, loss rate, and available bandwidth. Simulation results show that MPROBE can provide quality estimation with up to 90% average accuracy for all paths using only O($n\log n$) probes.

The remainder of the paper is organized as follows. Section 2 reviews related work. In Section 3, we present background information, formally define the overlay path probing problem, and describe a general approach to addressing it using inference. In Section 4, we introduce several algorithms for path selection and discuss the tradeoff between probing cost and estimation accuracy. We describe the performance evaluation results in Section 5, and draw conclusions in Section 6. Due to space limitations, many results of this study are omitted here, but can be found in [17].

## 2. Related work

Several methods have been proposed to reduce probing overhead in overlay networks [18]. In structured peer-to-peer systems, for example, each node maintains connections to only O($\log n$) [3] or O($n^{1/d}$) [4] neighbors, periodically

[1] Minimally Probing Routes in Overlays for Bounded Estimation.

probing them and replacing poorly connected ones as needed. Thus the overall probing overhead is reduced to O($n\log n$). In scalable application-level multicast systems such as NICE [2], nodes are organized in a hierarchy based on their distances to each other. Each node periodically selects a particular set of nodes to probe, in order to find its optimal location in the hierarchy. Usually the size of the probing set is a constant, so the overall probing overhead is O($n$). Although these methods can substantially reduce the total number of probes, as well as worst case link stress, probing "completeness" is sacrificed: only a small subset of the possible paths is probed in each round. Despite an effort to select promising paths for probing, it is still possible that many high-quality paths will remain unknown for long periods of time, especially in dynamic networks.

As with all the above approaches, MPROBE is designed to reduce probing overhead. However, instead of addressing the tradeoff between probing cost and probing completeness, MPROBE addresses the tradeoff between probing cost and probing accuracy. While other researchers recently have used linear equations to infer loss rate of unprobed overlay paths [19], they have addressed only additive metrics. In contrast, MPROBE is designed for both additive and bottleneck (namely, bandwidth) metrics. Since the overhead for bandwidth probing is typically much higher than that of latency and loss rate [20], MPROBE selects paths mainly for bandwidth inference optimization. In addition, we explicitly study the cost- completeness tradeoff on both additive and bottleneck metrics.

As we shall see, MPROBE uses the concept of set cover to compute bounded estimates for all overlay paths. A similar method has been studied in the context of physical networks [21]. In that approach, a minimal set of monitoring stations is identified such that all network links can be monitored using probes from these stations. The computation is based on greedy approximation algorithms to the set cover problem. A related network monitoring approach selects a minimum subset of paths to cover all links, and uses traceroute to obtain link-by-link latency of the selected paths [22]. The approach can then calculate all path latencies from the link latencies. MPROBE also uses set cover approximation algorithms to find a minimum set of paths whose links subsume all constituent physical links. The paths in the minimum cover are probed, and the results are used to obtain bounded estimations for all other paths. In contrast to the method in [21], which seeks a set of nodes for robust and efficient link monitoring, we investigate how the selection of paths affects estimation accuracy. Unlike the method in [22], MPROBE does not calculate latency of individual links, and thus it is independent of path probing techniques.

Finally, we note that the concept of inferring path quality from other probes is similar to network tomography methods that infer link delay [23], packet loss [24,25], or bottleneck bandwidth [26] from end-to-end measurements. However, tomography methods address quality inference for network links, while MPROBE addresses the quality of end-to-end overlay paths, where link sharing is an important consideration. In addition,since tomography focuses on inference accuracy, computationally intensive statistical methods, such as the EM algorithm [27], can be used to infer quality metrics. Since those algorithms are not practical for online quality estimation in large networks [28], however, MPROBE uses only simple arithmetic operations and comparisons, which are compatible with short probing periods of several seconds. Although the resulting estimates are less accurate, we emphasize that the required level of accuracy in overlay applications is not as high as in network tomography applications. Moreover, in MPROBE the accuracy is directly controllable by dispatching as many probes as needed.

## 3. Problem formulation and approach

In this section we formally define the overlay path probing problem and outline the MPROBE approach.

### 3.1. Concepts and definitions

An overlay network is a logical abstraction of the underlying physical network. Fig. 2 shows the composition of an overlay network, with the bottom layer representing the physical network, the top layer representing the overlay network, and the middle layer representing path segments (defined below). Formally, the overlay network is represented as a graph $G = \langle V, E\rangle$, where the vertex set $V$ is the collection of all overlay nodes, and the edge set $E$ is a subset of the $n \times (n-1)$ paths between overlay node pairs. All routers, as well as end nodes not involved in the overlay network, are abstracted away from the vertex set $V$. Similarly, the physical links of the selected overlay paths are hidden in the edge set $E$. We point out that while we actually model an overlay network as a directed graph, we use undirected graphs in examples to simplify the discussion.

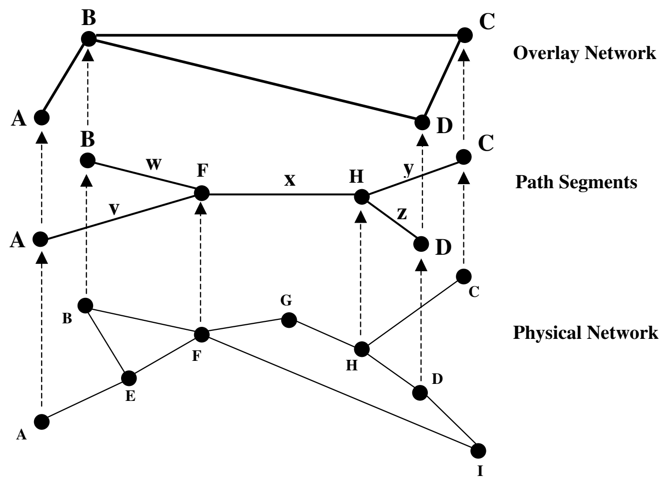


Fig. 2. An overlay network, showing path segments and physical network.

In sparse networks such as the Internet, overlay paths often intersect, that is, they share physical links. As shown in Fig. 2, path $AB = (AE, EF, FB)$ shares the physical links AE and EF with paths $AC = (AE, EF, FG, GH, HC)$ and $AD = (AE, EF, FG, GH, HD)$. To represent this property, we extend the overlay network model defined above to include *path segments* [29]. A path segment is a subpath of a physical path (i.e. network-level path) and comprises one or more physical links. Every overlay path comprises one or more path segments.

**Definition 1.** A *path segment* is a maximal subpath in an overlay network such that every inner vertex on the subpath is incident to exactly two physical links used in the overlay network.

Returning to Fig. 2, we see that the four edges of the overlay network actually comprise only five distinct path segments, labeled $v$, $w$, $x$, $y$ and $z$. The path segment set $S$ reflects link sharing among overlay paths. Using physical topology information, finding the path segment set for an overlay network is straightforward. The basic idea is to split a path or path segment into several subpaths until it shares no physical links with other paths or path segments.

We refer to an overlay network for which the path segment set $S$ is known, as a *topology-aware overlay network*, denoted $G_T = \langle V, S\rangle$. We will see the relevance of path segments shortly, but first let us formally define the overlay path probing problem.

### 3.2. The overlay path probing problem

Let $G_c = \langle V, E_c\rangle$ be the complete graph derived from the vertex set $V$. The construction of an overlay network is equivalent to the selection of a subset $E$ of $E_c$. This selection depends on the application-specific needs of the overlay network and the *quality* of the edges relative to those needs. To measure the quality of the overlay topology and the cost of the construction procedure, we need to define the following functions. The function $f(G)$ represents the overall quality of overlay network $G$, $Q(e)$ is the actual quality of edge $e$, and $Q'(e)$ is the *estimated* quality of $e$. The estimated quality of an edge can be measured directly via probing or can be inferred from the results of other probes. Let us assume the measured (probed) quality of an edge is equal to its actual quality, while an inferred value may be inaccurate. Further, let us assume that probing an edge $e$ incurs cost $C(e)$, while inferring the quality of an edge from other probes has no associated cost.

Under these assumptions, the overlay path probing problem is to make the overall estimated edge quality $Q'(E)$ as close as possible to the real quality $Q(E)$, while limiting the total probing cost. To measure the distance between $Q'(e)$ and $Q(e)$, we use the metric *estimation error*, $\delta$, which is the normalized error between a numeric value $x$ and its estimated value $x'$. Specifically, $\delta(x, x') = \frac{|x-x'|}{\max(x,x')}$. We note that the estimated quality of edge $e$ is a function of its actual quality $Q(e)$ and the set of edges (paths) $P \subseteq E_c$ selected for probing; formally $Q'(e) = h(Q(e), e, P)$. If $P$ contains $e$, then $Q'(e)$ is measured directly, else it must be inferred. In Section 4, we present several inference algorithms that realize the function $h$. We define *estimation accuracy* $Z$ as follows.

**Definition 2.** The estimation accuracy of path p under probing set P is $Z(p, P, Q) = 1 - \delta(Q(p), Q'(p)) = 1 - \delta(Q(p), h(Q(p), p, P))$.

**Definition 3.** The overall estimation accuracy under probing set P is $Z(P, Q) = \sum_i w_i Z(p_i, P, Q)$, a weighted sum[2] of estimation accuracy of all paths.

Given these definitions, we can formulate the overlay path probing problem as finding a set of edges $P \subseteq E_c$ and a function $h$, in order to maximize the overall estimation accuracy $Z(P, Q)$, subject to a cost constraint for probing edges in $P$: $C(P) = \sum_{e\in P} C(e) < K$, where $K$ is an application-specific limit. The complexity of this problem as well as its dual, minimizing $C(P)$ subject to $Z(P, Q) > K'$, depends on the correlation among $Q(e)$, $e \in E_c$. For example, if there is no link sharing among paths, then the correlation among $Q(e)$ is zero, and the problem has polynomial complexity. In many real environments, however, link sharing is common. Since it is possible to reduce the dual to the minimum set cover problem [30] in certain networks, this is a hard problem in general, and we propose heuristic solutions in this paper.

### 3.3. Role of path segments

Normally the estimated quality of paths is defined only for those paths that have been probed. Formally,

$$h(Q(e), e, P) = \begin{cases} Q(e), & \text{if } e \in P \\ \text{undefined}, & \text{otherwise} \end{cases}$$

Although we might *guess* the quality of the unprobed paths, this estimation is not bounded and is likely counterproductive to the construction of an optimal overlay network. Instead, we can exploit link sharing to reduce probing overhead by redefining $h(Q(e), e, P)$ for $e \notin P$. Specifically, the path segment set $S$ can be used to generate *bounded* quality estimates for *all* $n \times (n-1)$ paths, without the need for complete pair-wise probing. To see this, let us assume that the system probes only a subset of the paths. This action produces quality estimates (in terms of latency, loss rate, and available bandwidth) for each of the probed paths. Moreover, after probing a path, the quality of every *segment* in that path is bounded: (1) the latency of a path is greater than or equal to the latency of any of its subpaths; (2) the loss rate of a path is greater than or equal to the loss

[2] The weights are application specific, reflecting the relative importance of individual paths. In all our experiments, the weights are always equal.

rate of any of its subpaths; and (3) the available bandwidth of a path is less than or equal to the available bandwidth of any of its subpaths.

As an example, let us again consider the network in Fig. 2. If the measured loss rate of path AB is 5%, then the loss rate of neither segment $v$ nor segment $w$ can be higher than 5%. Since a path segment can be shared among multiple probed paths, it can have multiple sources of quality bounds. For example, if probing of path AC (comprising paths $v$-$x$-$y$) produces a measured loss rate of 3%, then we can lower the upper bound on the loss rate segment $v$ from 5% to 3%. Similarly, we can obtain an upper bound on latency and a lower bound on available bandwidth for every segment in each probed path. Moreover, as additional overlapping overlay paths are probed, the estimation accuracy for individual segments improves.

Now let us consider a path $p$ that is not directly probed. If the probing of *other* paths produced a bounded quality estimate for all of the segments in $p$, then we can generate a bounded estimate for the quality of the entire path $p$, based on slightly different observations: (1) the latency of a path is no greater than the sum of the latency upper bounds of its segments; (2) the loss rate of a path $p$ is no greater than $1 - \prod_{s \in p}(1 - r_s)$, where $r_s$ is the loss rate upper bound of segment $s$; and (3) the available bandwidth of a path is no lower than the minimum of the available bandwidth lower bound of its segments. Continuing our example using the network in Fig. 2, having probed paths AB and AC, we can make an assertion with high confidence that the loss rate of path BC is no greater than 11% $(1 - .95 \times .97 \times .97)$, even though BC was not probed. Hence, if the probing produces an estimate for all path segments in the overlay network, then we can compute bounded estimates for all $n \times (n - 1)$ paths.

### 3.4. General approach

The MPROBE solution to the overlay path probing problem described in this paper is centralized; in [31], we discuss a distributed version. In this centralized version, a node is elected as a *leader* and interacts with a topology server to maintain a minimal topology that covers all nodes and links involved in the overlay network. The leader is responsible for selecting paths for probing. It requests a node at one end of each path to probe the other end of the path. These nodes then return the probing results to the leader, which generates bounded quality estimations for all other paths. By adjusting the size of the probing set, the leader can trade probing overhead for estimation accuracy. The low computational overhead makes the approach suitable for online monitoring.

Although the leader node may appear to be a potential performance bottleneck, we will show that the per-round computational cost at the leader node is low in small- and medium-scale networks. Compared to pair-wise probing, MPROBE introduces only a small number of probing instruction packets at the leader. At other nodes, the computation and communication costs are greatly reduced relative to pair-wise probing.

Regarding the leader as a single point of failure, we assume that existing leader election and leader backup strategies can be used. Moreover, we point out that the leader model enables integration of MPROBE with hierarchical solutions [2], where overlay nodes are organized in clusters and a leader node is also selected for each cluster. For example, MPROBE can be applied in clusters where the topology data is available, the path overlap degree is high, and the traffic is dynamic, while other probing techniques might be applied in clusters lacking these properties.

Finally, since our concern in this paper is on quality estimation and the cost-quality tradeoff, we do not address single path measurement. An MPROBE implementation could adopt any of several proposed end-to-end techniques and tools [8] to measure latency, loss rate, and bandwidth. We adopt a coarse time scale for these metrics and assume they are stationary within one probing cycle. Therefore, different probing results can be combined in a cycle to infer the quality of other paths.

## 4. Path selection and quality estimation

In this section we propose several path selection and quality estimation algorithms.

### 4.1. Generic algorithm

We first describe a "generic" probing and estimation algorithm that realizes the inference technique described in Section 3.3. The generic algorithm comprises four main steps, as shown below.

(1) Select a set of unprobed overlay paths for probing.
(2) Issue probes on each of these paths and collect the results.
(3) Update the quality estimates of all segments contained in probed paths according to the following rules:
   (a) For the latency and loss rate metrics, the new result is assigned to those segments with a *greater* previous estimation value;
   (b) For the available bandwidth metric, the new value is assigned to those segments with a *lower* previous estimation value.
(4) Re-evaluate the quality of all affected paths. For each updated segment, all unprobed paths $p$ containing that segment are updated as follows:
   (a) For latency, sum the estimation values of all segments in $p$ and assign this sum as the new path latency estimate.
   (b) For loss rate, re-calculate $1 - \prod_{s \in p}(1 - r_s)$, where $r_s$ is the loss rate estimate of segment $s$, and assign this value as the new path loss rate estimate.
   (c) For available bandwidth, check if the updated estimation value of the particular segment is the

*lowest* among all segments in $p$. If so, then assign this value as the new estimate.

We refer to the inference method in steps 3 and 4 as *Minimax*.

### 4.2. Path selection

Let us now define several path selection algorithms, designed for different probing overhead metrics, for use in step 1 of the generic algorithm. Probing overhead can be represented as the number (or a weighted sum) of probes, the total physical link stress caused by probe traffic, or the worst-case link stress caused by probe traffic. The choice of metric is application specific.

The proposed algorithms use a two-phase strategy: (1) *set cover*, which selects paths to ensure bounded estimates for all paths; (2) *direct selection*, which selects additional paths whose probings are most likely able to refine the estimation of other paths. The number of additional paths selected in the second phase (which may be 0) is determined by whether: (1) the probing cost of the paths selected so far is lower than the cost limit $K$; (2) the marginal improvement on the overall estimation accuracy for each newly selected path is greater than an application-specific threshold value based on the historical performance data.

In the set cover phase, we need to use probes to find a valid estimate for every path segment in $S$, from which we can compute a quality estimation for all paths. Covering every segment with minimal probing overhead is an instance of the minimum set cover problem, where each path is a set, and the segments in a path are the elements of the set. We evaluate three different strategies:

(1) *NULL:* This is a degenerate case, where we actually skip the set cover phase. We eventually find a valid estimation for every path segment in the direct selection phase.
(2) *SETCOVER:* If we consider the *number* of probing packets as the probing overhead, then this is the standard set cover problem. We can use a greedy heuristic [30,32] to obtain the set cover that incurs minimum probing overhead. The basic idea is to choose, at each step, the path with maximum number of unprobed segments.
(3) *WSETCOVER:* If the probing of path $e$ incurs cost $C(e)$, then the problem corresponds to the weighted set cover problem, and the proposed heuristic [32] is to choose the path with the minimum ratio of its weight to the number of its unprobed segments.

In the direct selection phase, the path selection strategy is different from that of the set cover phase. In the set cover phase, we focus on finding a minimum cover: probing a segment that has already been probed is considered a waste of resources. In the direct selection phase, however, such probes can be used to refine the quality estimation of the relevant segments. Based on this observation, we propose five path selection strategies for this phase:

(1) *RANDOM:* Select paths randomly.
(2) *MINCOST:* Select the unprobed path with the lowest cost. In this study, the probing cost is defined as the number of *physical hops*.
(3) *MINSEG:* Simply select the path with the least number of *segments*, a special case of strategy (2). The probing cost of a path in this study is defined as proportional to the number of its constituent segments, since a path with fewer segments produces a more accurate estimate.
(4) *MAXREF:* Select the path with the maximum reference number, defined to be the number of other unprobed paths sharing links with it. The probing result for a path that shares links with other unprobed paths will improve the estimation accuracy of those paths.
(5) *MINSTRESS:* Select the path that has the lowest link stress due to probes of already selected paths. The link stress of a path is the maximum link stress among its segments. The goal of this strategy is to evenly distribute probing stress across links, so that the probability of improving the quality estimation is approximately the same across all segments. In this strategy as well as in *MAXREF*, the selection of one path affects the selection of subsequent paths, in contrast to the first three strategies.

In summary, we propose three set cover strategies and five direct selection strategies. The resulting 15 algorithms are listed in Table 1. The results presented later show that no single algorithm is best suited for all situations. Rather, the algorithm should be selected based on the network topology and the application requirements.

Let us use an example in the small network in Fig. 2 to demonstrate the operation and performance of these algorithms. The overlay network comprises four nodes: A, B, C and D. (Again, for simplicity we assume the edges are undirected.) Let the available bandwidths of path segments $v$, $w$, $x$, $y$, $z$ be 4, 1, 100, 3, 2 Mbps, respectively. So, the actual available bandwidth on the six paths AB, AC, AD, BC, BD, CD is 1, 3, 2, 1, 1, and 2, respectively. Table 2 lists *all* possible probing sets selected from among the six paths. Each entry includes the paths probed, the corresponding estimation accuracy, and maximum link stress produced by the probes. For example, after probing path AB, the estimation accuracy for AB is 1.0, since the bandwidth determined by the probe matches the exact bandwidth of AB. The estimation accuracy for the other five paths is 0, however, because each contains at least one path segment for which the single AB probe provides no information. So the overall estimation bandwidth for this entry is $1/6 \approx 0.17$. Similarly we can calculate the overall estimation accuracy for other probe sets. Entries marked with a $*$ in Table 2 are optimal in both the estimation accuracy and the maximum link stress produced.

Table 1
Path selection algorithms

| Direct selection strategies | Set cover strategies | | |
|---|---|---|---|
| | NULL | SETCOVER (minimize number of probes) | WSETCOVER (minimize total probing cost $\sum_e C(e)$) |
| RANDOM | NULL_RANDOM | SETCOVER_RANDOM | WSETCOVER_RANDOM |
| MINCOST | NULL_MINCOST | SETCOVER_MINCOST | WSETCOVER_MINCOST |
| MINSEG | NULL_MINSEG | SETCOVER_MINSEG | WSETCOVER_MINSEG |
| MAXREF | NULL_MAXREF | SETCOVER_MAXREF | WSETCOVER_MAXREF |
| MINSTRESS | NULL_MINSTRESS | SETCOVER_MINSTRESS | WSETCOVER_MINSTRESS |

As an example of algorithm performance, we consider the execution of the SETCOVER_MINSTRESS algorithm corresponding to the following entries: (AC, 0.17, 1), (AC, BD, 0.83, 2), (AC, BD, AB, 0.83, 2), (AC, BD, AB, CD, 1, 2), (AC, BD, AB, CD, AD, 1, 3), and (AC, BD, AB,! CD, AD, BC, 1, 4). After the first two probes (AC and BD), all segments are covered, and the other segments are added under the direct selection phase. We note that for all cases except three probes, both the estimation accuracy and the maximum link stress are optimal. As another example, executing SETCOVER_RANDOM with a maximum of three probes often produces an optimal value of estimation accuracy (1.0); in a set of ten experimental runs, for instance, the algorithm found this value 4 times, with an average accuracy at 0.9 and maximum link stress of 2.6. These examples show the proposed algorithms can obtain optimal or near-optimal solutions.

## 4.3. Improving latency and loss rate estimation

Next, we describe a technique that can be used to improve estimates on additive metrics (latency and loss rate). There exists a fundamental difference between latency estimation and bandwidth estimation. The estimation of a segment's available bandwidth may be equal to its real value. Probing additional paths containing a particular segment increases the chance to obtain the real available bandwidth of the segment. However, the latency of a segment is calculated as the minimum latency of the paths that include this segment. In other words, the latency estimation of a segment is a sum (instead of maximum or minimum) of the segment's real latency and the latency of other segments. Therefore, unless a probed path comprises only one segment, the estimate will never be equal to the real segment latency.

To address this issue, we extend the proposed methods with an enhancement based on an algebraic approach proposed by Shavitt et al. [29]. This method exploits the cumulative property of path latency to produce more accurate estimations. As an example, let us assume the measured latency of path AB in Fig. 2 is a constant $L_{\mathrm{AB}}$, the latency of AC is $L_{\mathrm{AC}}$, and the latency of CD is $L_{\mathrm{CD}}$. Let $L(e)$ be the latency estimation of path $e$ or segment $e$. Then we have the following linear equations:

$$\begin{cases} L(v) + L(w) = L_{\mathrm{AB}}, \\ L(v) + L(x) + L(y) = L_{\mathrm{AC}}, \\ L(\mathrm{BC}) = L(w) + L(x) + L(y), \\ L(\mathrm{AD}) = L(v) + L(x) + L(z), \\ L(y) + L(z) = L_{\mathrm{CD}}, \\ L(\mathrm{BD}) = L(w) + L(x) + L(z) \end{cases}$$

There exists an infinite number of solutions for $L(\mathrm{BC})$, $L(\mathrm{AD})$, and $L(\mathrm{BD})$ that satisfy these constraints. However, if we know the probing results $L(\mathrm{BC})$ and $L(\mathrm{AD})$, then we can solve all the linear equations. Generally, if we can find $|S|$ paths to probe, and the resulting linear equations correspond to a $|S| \times |S|$ non-singular matrix, then we can solve all $|S|$ linear equations and obtain the real latency of all segments in $|S|$, from which we can derive the latency of all unprobed paths. The power of this approach is that, usually $|S| \ll n^2$, which implies that we can obtain accurate latency estimations for all $n \times (n-1)$ paths with relatively few probes.

In many cases, the rank of the matrix may be less than $|S|$. In this case, we cannot solve for all the variables. However, our ultimate goal is to estimate the quality of paths, not segments. A rank-deficient matrix indicates that some equations can be expressed as linear combinations of other equations. For example, we can express $L(\mathrm{BD}) = L(w) + L(x) + L(z) = L(\mathrm{AD}) + L(\mathrm{BC}) - L_{\mathrm{AC}}$ to calculate the quality of path BD without solving the equations. Simulation results presented in Section 5 show the savings can be substantial. Moreover, we can use the same strategy on loss rate estimation. Assume the loss rate of path $p$ is $r_p = 1 - \prod_{s \in p}(1 - r_s)$, where $s$ is a segment of path $p$. By defining $R = \log(1 - r)$, we can convert the loss rate observation to a linear equation: $R_p = \sum_{s \in p} R_s$.

Based on this analysis, we propose an improved strategy in Fig. 3, called *Minimax-Algebraic*, for path selection and quality inference. Since MPROBE is a common probing approach for both additive and bottleneck metrics, we use the algorithms in Table 1 as the primary method for path selection, applying the algebraic method only for minor adjustment. In the last two steps of the generic algorithm, the algebraic method is used to complement *Minimax* to obtain accurate estimates for latency and loss rate of unprobed paths.

Table 2
Overall estimation accuracy and maximum link stress for all possible probing sets in the example

| 1 Probe | 2 Probes | | 3 Probes | | 4 Probes | | 5–6 Probes |
|---|---|---|---|---|---|---|---|
| (AB, 0.17, 1)* | (AB, AC, 0.5, 2) | (AB, AD, 0.5, 2) | (AB, AC, AD, 1, 3) | (AB, AC, BC, 0.5, 2) | (AB, AC, AD, BC, 1, 3) | (AB, AC, AD, BD, 1, 3) | (AB, AC, AD, BC, BD, 1, 4) |
| (AC, 0.17, 1)* | (AB, BC, 0.39, 2) | (AB, BD, 0.42, 2) | (AB, AC, BD, 0.83, 2) | (AB, AC, CD, 1, 2)* | (AB, AC, AD, CD, 1, 3) | (AB, AC, BC, BD, 0.83, 3) | (AB, AC, AD, BC, CD, 1, 3)* |
| (AD, 0.17, 1)* | (AB, CD, 0.33, 1) | (AC, AD, 0.5, 2) | (AB, AD, BC, 0.81, 2) | (AB, AD, BD, 0.5, 2) | (AB, AC, BC, CD, 1, 3) | (AB, AC, BD, CD, 1, 2)* | (AB, AC, AD, BD, CD, 1, 3)* |
| (BC, 0.17, 1)* | (AC, BC, 0.5, 2) | (AC, BD, 0.83, 2)* | (AB, AD, CD, 0.94, 2) | (AB, BC, BD, 0.72, 3) | (AB, AD, BC, BD, 0.81, 3) | (AB, AD, BC, CD, 0.94, 2) | (AB, AC, BC, BD, CD, 1, 3)* |
| (BD, 0.17, 1)* | (AC, CD, 0.5, 2) | (AD, BC, 0.81, 2) | (AB, BC, CD, 0.81, 2) | (AB, BD, CD, 0.81, 2) | (AB, AD, BD, CD, 0.94, 3) | (AB, BC, BD, CD, 0.81, 3) | (AB, AD, BC, BD, CD, 0.94, 3) |
| (CD, 0.17, 1)* | (AD, BD, 0.5, 2) | (AD, CD, 0.44, 2) | (AC, AD, BC, 1, 3) | (AC, AD, BD, 1, 3) | (AC, AD, BC, BD, 1, 4) | (AC, AD, BC, CD, 1, 3) | (AC, AD, BC, BD, CD, 1, 4) |
| | (BC, BD, 0.42, 2) | (BC, CD, 0.5, 2) | (AC, AD, CD, 0.5, 2) | (AC, BC, BD, 0.83, 3) | (AC, AD, BD, CD, 1, 3) | (AC, BC, BD, CD, 1, 3) | (AB, AC, AD, BC, BD, CD, 1, 4)* |
| | (BD, CD, 0.5, 2) | | (AC, BC, CD, 1, 3) | (AC, BD, CD, 1, 2)* | (AD, BC, BD, CD, 0.94, 3) | | |
| | | | (AD, BC, BD, 0.81, 3) | (AD, BC, CD, 0.94, 2) | | | |
| | | | (AD, BD, CD, 0.94, 3) | (BC, BD, CD, 0.5, 2) | | | |

Using Gaussian elimination to check linear dependency of segment vectors, this approach requires $O(|E_c| \cdot |P|)$ computational steps. As we shall see in Section 5, this approach is very effective in improving accuracy of estimates while limiting probing costs.

## 5. Performance evaluation

In this section we present the performance evaluation results for the MPROBE algorithms. First we introduce the simulation environments. Second, we evaluate the cost-accuracy tradeoff for a baseline algorithm, demonstrating that approximately $n \log n$ probes are sufficient to find bounded estimates for all paths. In the case of bandwidth, the quality (accuracy) of these estimates is very high. Third, we compare the performance of different path selection algorithms. Fourth, we present the performance results for enhanced algorithms that use the algebraic method. Finally, we study the effect of incomplete topology information on algorithm performance.

### 5.1. Simulation setup

We have evaluated the MPROBE algorithms on six different physical network topologies, including a real AS-level Internet topology [10], three generated by the GT-ITM topology generator [33], and two generated by Inet3.0 [34]. The parameters and additional description of these topologies are presented in Table 3. All these topologies are Internet-like topologies except for “r1000”, which is intended to support the evaluation of MPROBE on dense topologies.

We vary the size of the overlay network from 4 to 256 nodes. The overlay nodes are uniformly selected from the nodes in underlying physical topologies. We assume the IP layer uses shortest path routing with generated weight (for GT-ITM and Inet topologies) or number of physical hops (for the “as6474” topology) as the metric. In the path selection algorithms the weight of a path is set to the number of physical hops between the two end nodes. We execute the probing algorithms on each configuration 10 times with different random seeds and report the average results. All data ranges are available in [17]; some of them are plotted in the figures as error intervals.

We vary the delay for backbone links from 1 to 50 ms. The delay on links from edge routers to end hosts is randomly set between 1 and 3 ms. We use the LM1 model [24] for spatial distribution of backbone link loss rate, where the good link fraction, $f$, is set to 90%. The loss rate on a “good” link is between 0 and 1%, and the loss rate of a “bad” link is between 5% and 10%. For edge links, the good link fraction is 50%, the “good” link loss rate is set between 0 and 1%, and the “bad” link loss rate is between 10% and 20%. We randomly set available bandwidth between 100 to 500 MB for backbone links, and 500 KB to 1 MB for edge links.

```
1 ) $P := \emptyset$;
2 ) $Q := E_c$;
3 ) while $C(P) < K$ and $Q \neq \emptyset$ do begin
4 )     select a path $p$ from $Q$ using one of the path selection algorithms;
5 )     if the segment vector of $p$ is not linearly dependent on any subset of paths in $P$ then
6 )         add $p$ to $P$;
7 )     remove $p$ from $Q$;
8 ) end
9 ) probe the selected paths in $P$;
10) use the Minimax method in the generic algorithm to derive segment quality estimates;
11) if any subset of the linear equations derived from paths in $P$ can be solved then
12)     solve the equations to obtain exact latency (loss rate) values for a subset of segments;
13) forall paths $p$ in $E_c - P$ do begin
14)     compute path quality of $p$ from segment quality values using Minimax;
15)     if the segment vector of $p$ is linearly dependent on a subset of paths in $P$ then
16)         derive path latency (loss rate) of $p$ from those paths;
17) end
```

Fig. 3. The improved approach for path selection and quality inference (*Minimax-Algebraic*).

Table 3
Topologies used in evaluation

| Topology name | Vertices | Links | Description |
|---|---|---|---|
| as6474 | 6474 | 12,572 | Real Internet AS-level topology as of February 2000 |
| inet6474 | 6474 | 11,911 | Generated by Inet3.0, simulated AS-level topology as of February 2000 |
| inet12700 | 12,700 | 28,121 | Generated by Inet3.0, simulated AS-level topology as of February 2002 |
| r1000 | 1000 | 36,443 | GT-ITM flat topology |
| ts1000 | 1000 | 2048 | GT-ITM transit-stub topology |
| ts10000 | 10,000 | 23,238 | GT-ITM transit-stub topology |

## 5.2. Baseline results

We first evaluate the basic cost-accuracy tradeoff for the SETCOVER_RANDOM algorithm, which we regard as the "baseline" version MPROBE. Results of other algorithms are presented in Section 5.3. We set the estimation weights of all segments to be equal, so that the overall estimation accuracy is equivalent to the average estimation accuracy.

### 5.2.1. Probing cost

Probing cost is proportional to estimation accuracy: the more accurate the estimation, the more probes required. In the extreme case, we need all $n \times (n-1)$ probes for a 100% accurate estimation. Here we examine the probing cost for specified levels of overall estimation accuracy.

In Fig. 4, we plot the number of probes needed versus network size for several estimation accuracy levels. Fig. 4a is for latency estimation, Fig. 4b is for loss rate estimation, and Fig. 4c is for bandwidth estimation. The underlying topology is "as6474." The network size varies from 4 to 256. First let us consider the minimum number of probes needed by MPROBE to produce bounded estimation for all paths. This number corresponds to the size of a minimum set cover. The curves in Fig. 4 marked as "All-Bounded" show that this number is approximately O($n \log n$). This result implies we can save many probes if only bounded estimates are required (for example, bounded latency implies the path is not disconnected).

Next we evaluate the number of probes needed for overall estimation accuracy levels of 50%, 70%, 80%, and 90%, for each of the three metrics. Fig. 4c shows that for bandwidth the accuracy of a minimum set cover (labeled "All-Bounded") is usually close to 90%. Hence, even with a small number of probes, the estimates are quite accurate.

For latency, however, the number of probes required to achieve the same level of accuracy is much higher than for bandwidth, as shown in Fig. 4a. This result is due to the fact that a path latency estimation is a *sum* of the minimum latency estimation of several other paths, and therefore less accurate. As we shall see later, the improved latency estimation algorithm effectively solves this problem.

Fig. 4b shows that the performance of the SETCOVER_RANDOM algorithm on loss rate is between that of bandwidth and latency. Although the loss rate estimation strategy is also based on summation, the variance in loss rate among paths in these simulations, as well as that in real Internet environments, is smaller than the variance in path latency. Again, we will see later that the algebraic method is effective in addressing this issue.

Now let us evaluate the performance of the SETCOVER_RANDOM algorithm on other topologies. Fig. 5 plots the number of probes needed for an overall estimation accuracy level of 90% for the three metrics. The results for other estimation accuracy levels are given in [17]. In the figure, the topology names are shown on the top, and the numerical values shown below the data points are for bandwidth. The performance for bandwidth differs slightly among the topologies. Generally, larger (such as "inet12700") and denser (such as "r1000") topologies diffuse overlay paths and provide less space for the accuracy-overhead tradeoff. However, even on the densest topology ("r1000", with average node degree 72.8!), the baseline SETCOVER_RANDOM algorithm still achieves 90% accuracy for bandwidth estimation using only O($n \log n$) probes.

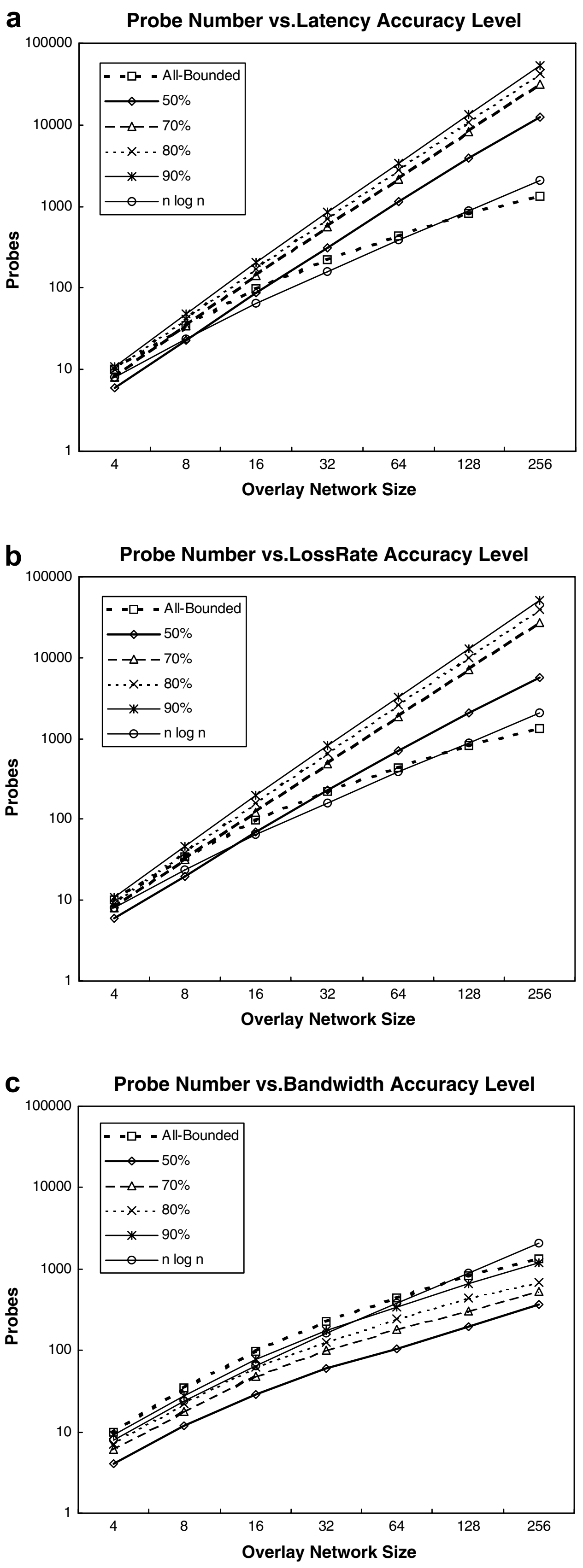


Fig. 4. Minimum number of probes for given levels of estimation accuracy (SETCOVER_RANDOM, "as6474"). (a) Latency accuracy level, (b) loss rate accuracy level, and (c) bandwidth accuracy level.

Fig. 6 shows the number of probes needed, as well as the corresponding overall estimation accuracy, for a minimum set cover on different topologies. The number of probes is shown on the top plot, while the estimation accuracy on the bottom plot. The dashed lines on the top plot indicate the value of $n \log n$. Again, the number of probes needed for a minimum set cover is $O(n \log n)$ on all six topologies (GT-ITM topologies have the lowest values), while the corresponding accuracy values for bandwidth estimation are all close to or above 90%.

Figs. 5 and 6 show the performance of the SETCOVER_RANDOM algorithm on the "as6474" topology is representative. Considering this fact and that the "as6474" topology is the only real Internet topology used in the simulations, we focus on this topology in the simulation results presented in remainder of the paper. Additional results can be found in [17].

### 5.2.2. Estimation accuracy

In this subsection we assess the estimation accuracy for different numbers of probes. We present both the distribution and the average of the estimation accuracy of the SETCOVER_RANDOM algorithm.

Figs. 7a–c plot the distribution of estimation accuracy versus number of probes for latency, loss rate, and bandwidth, respectively. As shown in Fig. 7c, for bandwidth the number of inaccurate estimations (those with accuracy value lower than 1.0) drops quickly as the number of probes increases. In other words, a relatively small number of probes produces exact values (accuracy value of 1.0) for most paths. These figures confirm the implications of the last subsection, namely, that later probes are generally less useful than the earlier probes, and that MPROBE can produce reasonably accurate estimates by probing only a small fraction of the total paths.

Figs. 7a and b show again that the latency and loss rate estimations are less accurate than that of bandwidth. Indeed, in both cases number of paths with 100% accuracy is exactly the same as the number of probes. This behavior implies that all the accurate latency and loss rate values are obtained by direct probing instead of by estimation, and thus none of the inferred estimates are 100% accurate for the baseline SETCOVER_RANDOM algorithm. In addition, both figures show a large number of paths still have low accuracy (less than 0.3) even after a substantial number of probes. Fig. 8 plots the maximum (always 1.0), average, and minimum estimation accuracy for latency, loss rate, and bandwidth.

## 5.3. Algorithm comparison

Next we compare the performance of the path selection algorithms in terms of probing cost and estimation accuracy. Fig. 9 evaluates different approaches to the first (set cover) phase of path selection. Figs. 9a–c compare the average estimation accuracy of three random algorithms (NULL_RANDOM, SETCOVER_RANDOM, and

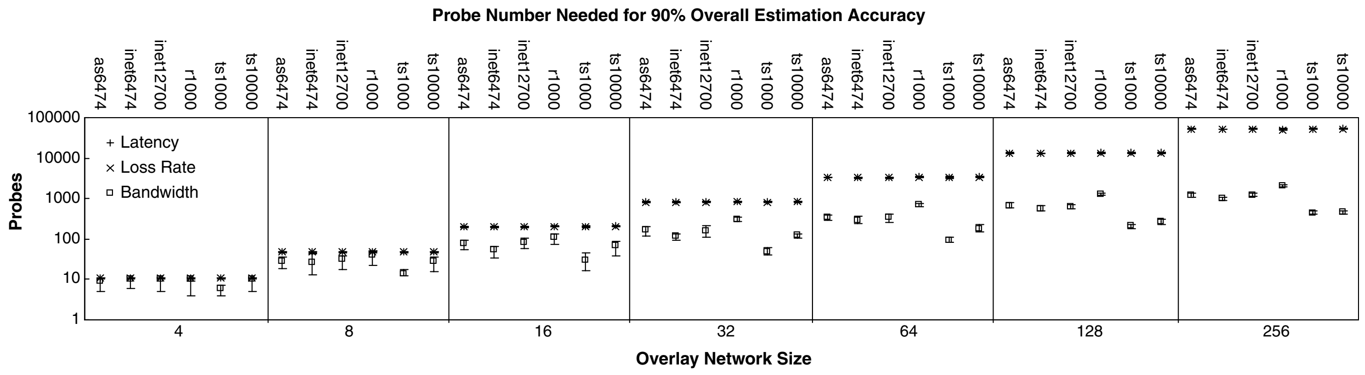


Fig. 5. Minimum number of probes (with error intervals) for 90% estimation accuracy.

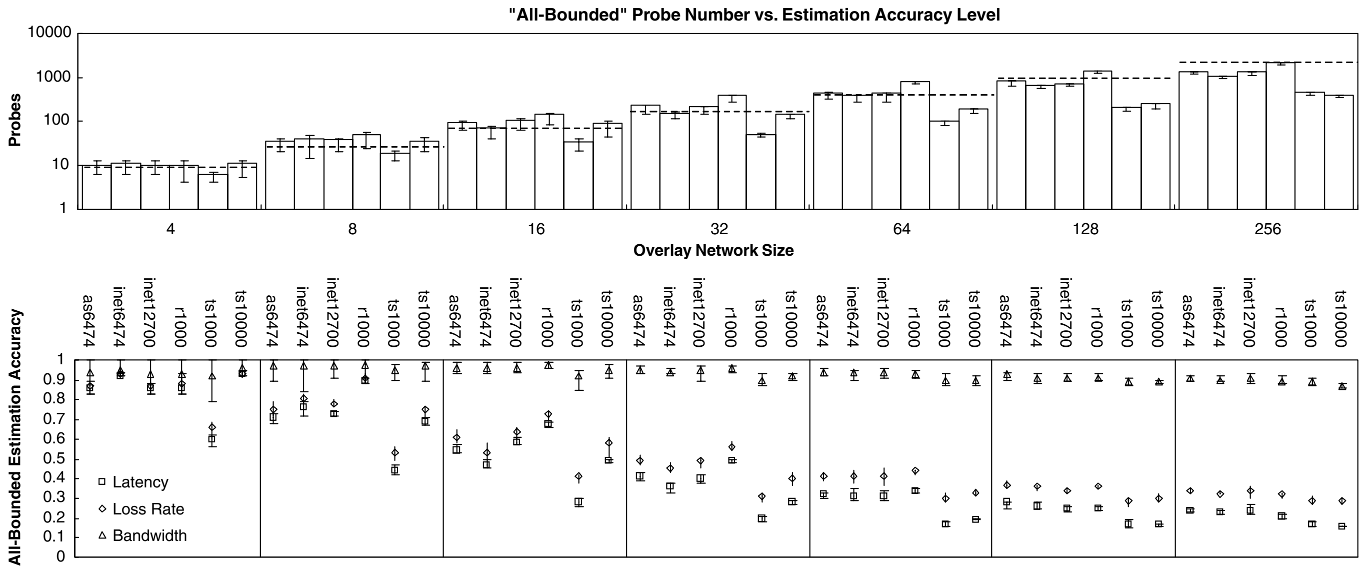


Fig. 6. Minimum number of probes and the corresponding estimation accuracy (with error intervals) for minimum set covers.

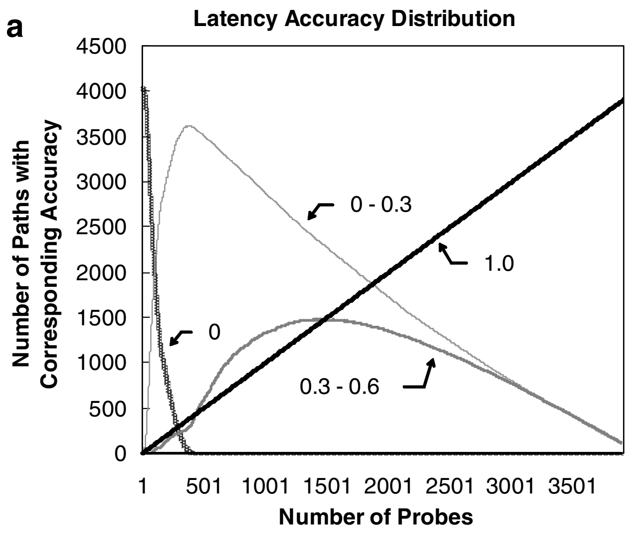
WSETCOVER_RANDOM) on latency, loss rate, and bandwidth, respectively. As expected, the algorithms with set cover strategies (SETCOVER_RANDOM and WSETCOVER_RANDOM) have an advantage over NULL_RANDOM. The improvement is larger for bandwidth estimation accuracy (up to 0.2) than for latency and loss rate. The weighted set cover algorithm performs slightly worse than the minimum set cover algorithm, since the former is designed to minimize total link stress instead of number of probes. As shown in [17]., the difference of estimation accuracy in the second (direct selection) phase of the algorithms is small.

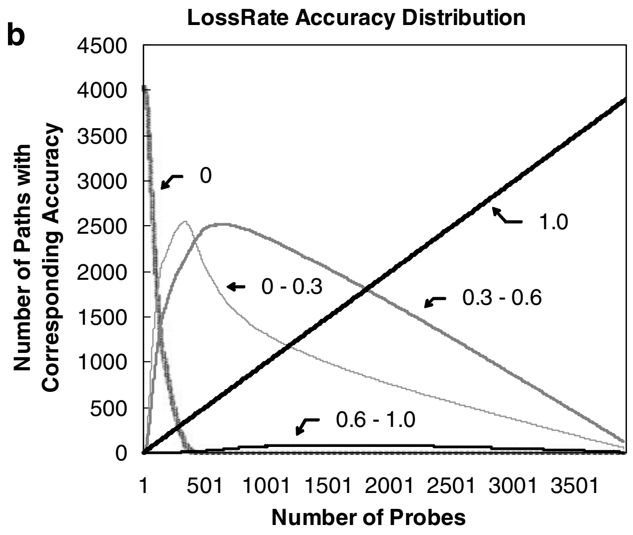
Comparisons in terms of link stress due to probing can be found in [17]. Generally, the weighted set cover algorithms perform slightly better than their set cover counterparts in average link stress, since they select paths with low probing cost, usually producing shorter paths in terms of physical hops. The difference, however, is small. Simulation results in [17] indicate that the choice of direct selection strategies has more effect on link stress than does the method used in the set cover phase: the SETCOVER_MINSTRESS algorithm exhibits lower maximum link stress than the other algorithms.

Combined, these results suggest that, depending on the path quality and probing overhead definition, the algorithms SETCOVER_RANDOM and SETCOVER_MINSTRESS are more efficient than the other algorithms in terms of low cost-accuracy ratio and low congestion.

## 5.4. Improved latency and loss rate estimation

Now let us evaluate the effect on performance of using algebraic methods in estimating latency and loss rate. As shown in Figs. 10a and b, this approach (*Minimax-Algebraic*) can significantly improve the performance for both latency and loss rate estimation. In both cases, we see there exists a threshold point, below which the original algorithm (SETCOVER_RANDOM) performs better than the pure algebraic method (without using *Minimax*). After this point, the algebraic method quickly derives the quality of all paths. The *Minimax-Algebraic* approach achieves optimality in all cases.





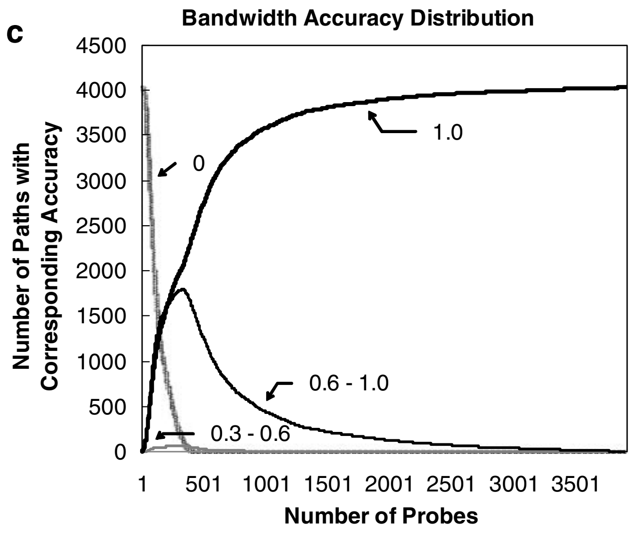
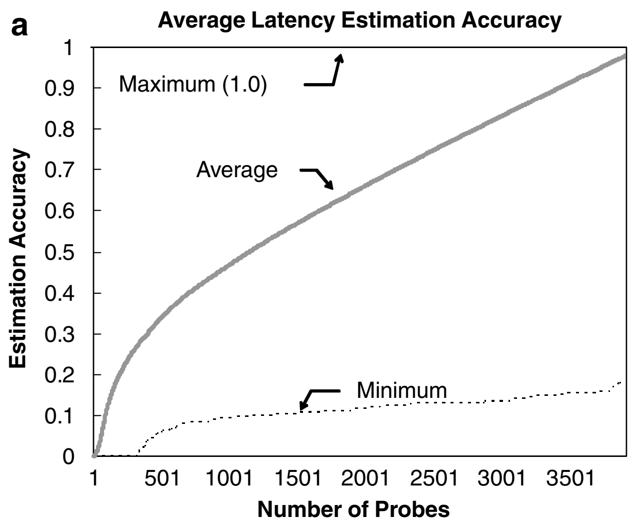
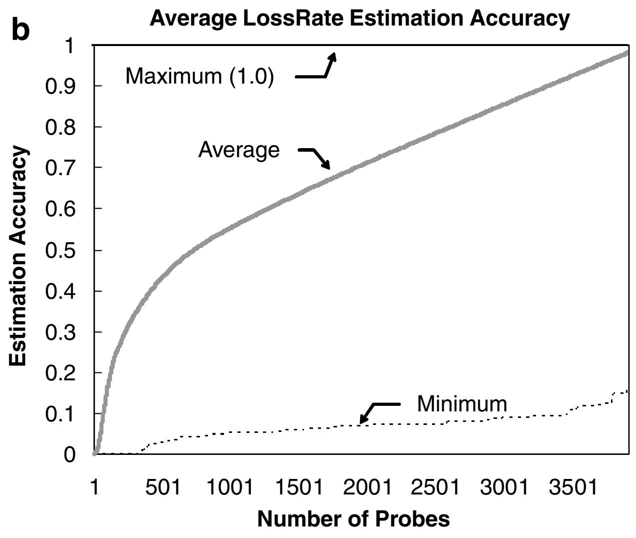


Fig. 7. The estimation accuracy distribution versus number of probes (SETCOVER_RANDOM, "as6474", 64 nodes). (a) Latency accuracy distribution, (b) loss rate accuracy distribution, and (c) bandwidth accuracy distribution.





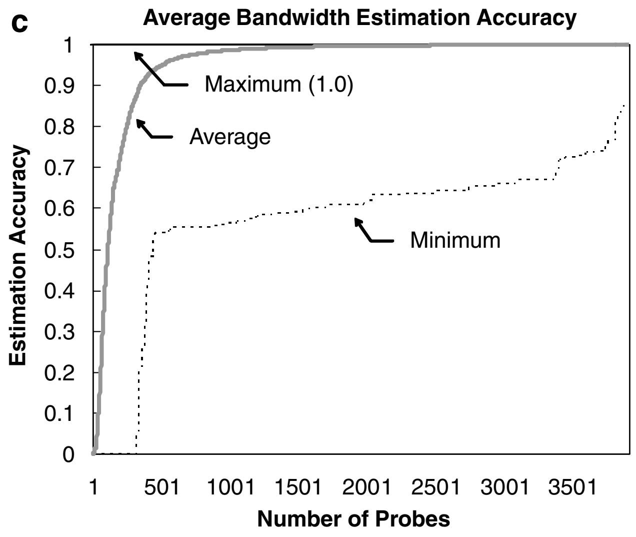
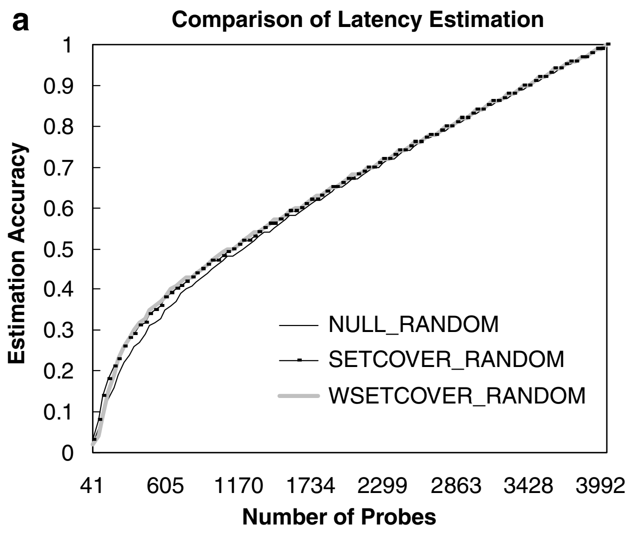
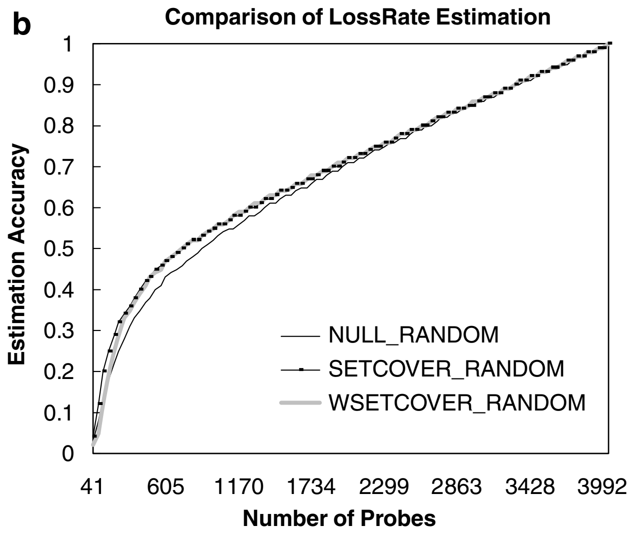


Fig. 8. The average estimation accuracy (SETCOVER_RANDOM, "as6474", 64 nodes). (a) Averge latency estimation accuracy, (b) average loss rate estimation accuracy, and (c) average bandwidth estimation accuracy.

a


b


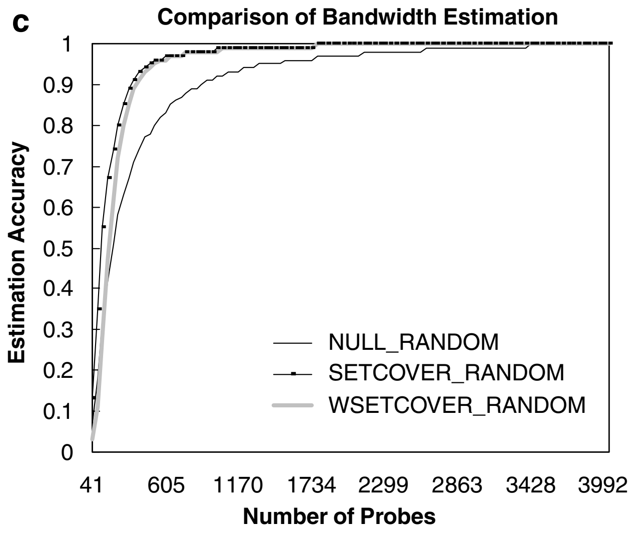
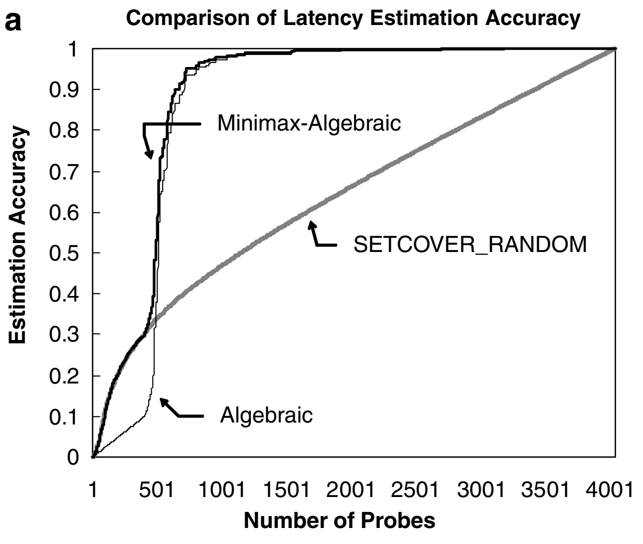
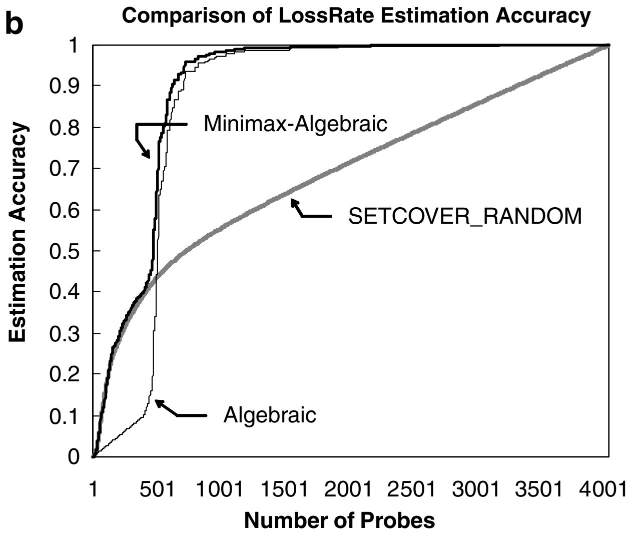
c


Fig. 9. Comparison of random path selection algorithms (“as6474”, 64 nodes). (a) Latency estimation, (b) loss rate estimation, and (c) bandwidth estimation.

a


b


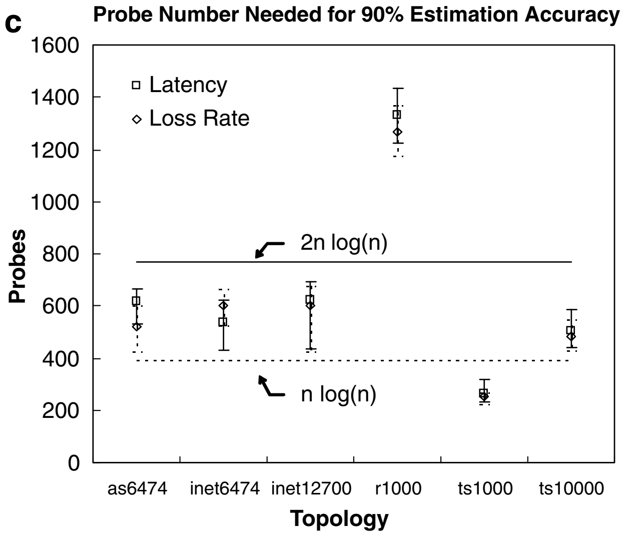
c


Fig. 10. Improved estimation algorithm results (64 nodes). (a) Latency estimation accuracy, (b) comparison of loss rate estimation accuracy, and (c) probe number needed for 90% estimation accuracy.

Fig. 10c plots the number of probes needed for 90% estimation accuracy for latency and loss rate on all six topologies. The results implies, by using *Minimax-Algebraic*, we can achieve 90% accuracy for latency and loss rate estimation using only O($n\log n$) probes, as in the case of bandwidth estimation. The results for other accuracy levels are given in [17].

### 5.5. The effect of incomplete topology information

Lastly, we address the effect of incomplete topology information on the performance of MPROBE. In a real Internet environment, topology information available to end nodes may be incomplete or inaccurate. For example, if *traceroute* is used to gather such data, Internet routers that do not respond to ICMP ECHO messages will not appear in the corresponding path composition information. In our simulations, we adjust the percentage of these "unresponsive" routers, thereby varying, the degree of topology incompleteness.

Considering the inference methods described in Section 3, if some vertices on a path are missing, then MPROBE may be unaware of corresponding link sharing, and thus unable to exploit it. For example, in a topology-aware overlay network shown on the left in Fig. 11, quality lower bounds for paths AB, AD, BC, CD can be inferred by probing paths AC and BD. If router $R_1$ becomes unresponsive, then the topology-aware overlay network will change, as shown on the right in Fig. 11. In this case, MPROBE may be unable to infer any quality bound for path AB from the same probing results. However, MPROBE can still obtain valid estimates for paths AD, BC, and CD. Hence, while topology incompleteness does not affect the *correctness* of the inference methods, the degradation in the *quality* of estimations depends on the amount of missing information. Let us evaluate this effect.

Fig. 12 plots the estimation accuracy when the unresponsive router rate is set to 1%, 5%, 10%, 20%, 30%, and 50%. As shown, the performance degradation is tolerable for low unresponsive rates. Even with 20% of the routers not responding, the performance of SETCOVER_RANDOM algorithm for bandwidth and the improved estimation algorithm for latency and loss rate produce results that are comparable to the original performance of NULL_RANDOM algorithm. Moreover, despite the drop in performance when the unresponsive rate is higher than 20%, the MPROBE algorithms still provide valid, albeit looser, quality bounds. In addition, MPROBE will quickly generate more accurate inferences when more complete topology information is available, for example, via a topology server.

### 5.6. Discussion

The MPROBE approach proposed in this paper is based on the assumption that sharing of physical links among overlay paths is substantial. If the degree of sharing is small, then MPROBE will provide little or no benefit. However, in this case, the worst-case link stress is likely to be low even for a pair-wise probing. On the other hand, in sparse Internet-like networks such as those used in the above simulations, link sharing is common and can be exploited by the proposed inference techniques.

MPROBE is most suitable for applications that require only bounded, rather than precise, estimations for all paths. By setting the probe number to approximately $2n\log n$ and using the improved *Minimax-Algebraic* approach, MPROBE can achieve 90% estimation accuracy for both additive and bottleneck metrics, which indicates it may be useful to a broad range of overlay applications.

Another strategy to improve the estimation accuracy is to exploit historical path quality data. For example, the path selection algorithm might always choose paths with low latency, low loss rate and high bandwidth, which will have more effect on the estimation accuracy for other paths. Specifically, since a probe on high quality path can generate high quality estimation for segments in that path, it may improve quality estimations for intersecting paths. In [17], we discuss two additional path selection algorithms that use such historical data. However, our results show

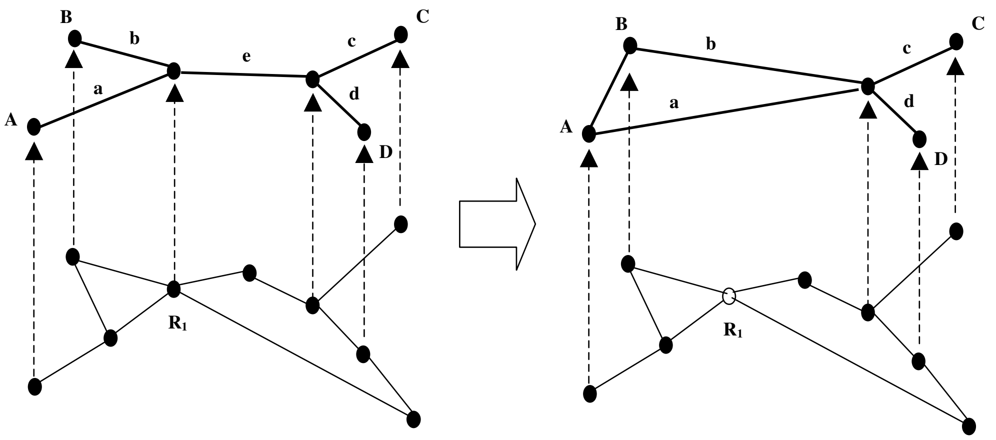


Fig. 11. The effect of incomplete topology information on overlay network.

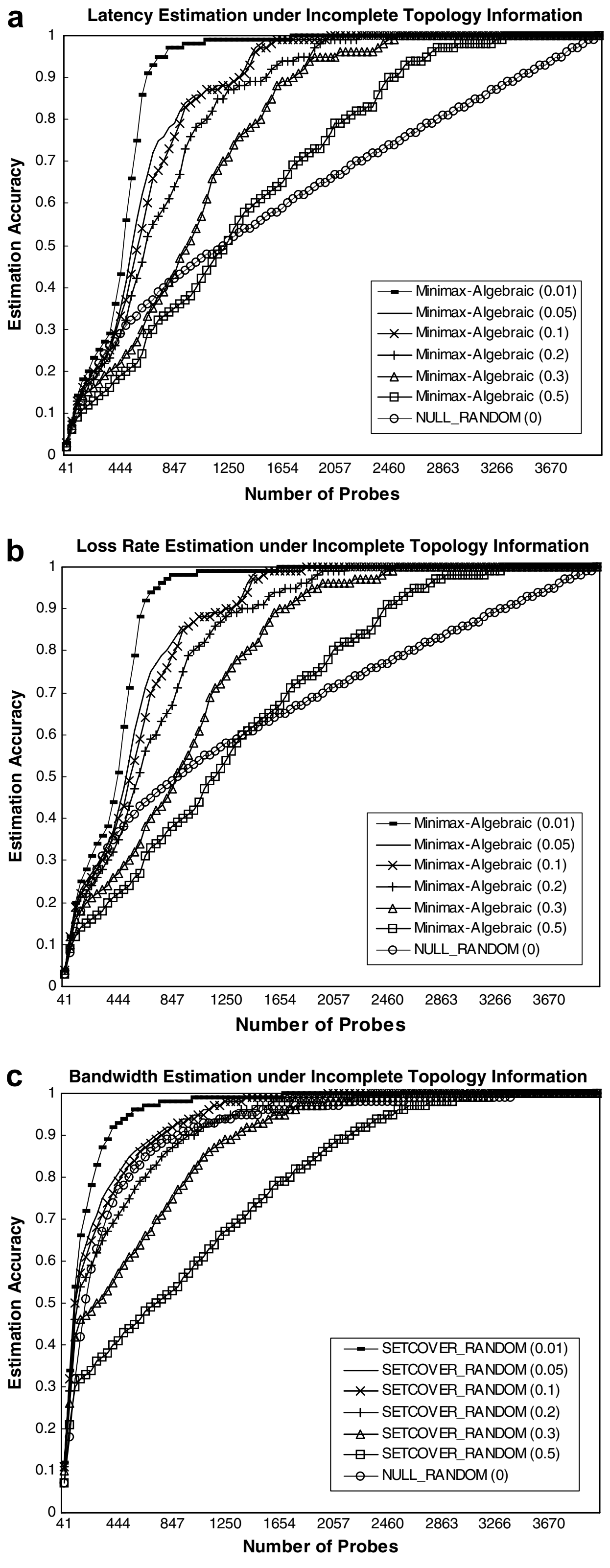


Fig. 12. The average estimation accuracy on incomplete topology information ("as6474", 64 nodes). (a) Latency estimation under incomplete topology information, (b) loss rate estimation under incomplete topology information, and (c) bandwidth estimation under incomplete topology information.

these history-based methods offer only marginal improvement, which must be weighed against their relatively high complexity.

## 6. Conclusions

In this paper we proposed and evaluated MPROBE, an overlay path probing approach that can trade probing overhead for accuracy of path quality estimation. The approach exploits link sharing among overlay paths, enabling a probing result for one path to reveal partial quality information for other paths. MPROBE uses this property to compute bounded estimates for unprobed paths. The quality metrics include latency, loss rate and available bandwidth. We designed several path selection algorithms and compared them in terms of probing cost and estimation accuracy. Simulation results show that, depending on the topology, MPROBE can provide quality estimations with up to 90% average accuracy for all $n \times (n-1)$ overlay paths using only $O(n \log n)$ probes.

Although this approach uses physical topology information to infer path qualities, it does not require this information to be complete; estimation accuracy gracefully degrades with the amount of missing information. Simulation results show this approach can maintain good performance even when 20% of the topology information is not available. We argue this property makes the approach practical in Internet environments, and experimental studies on the PlanetLab Internet testbed [11] support this conclusion. We anticipate that MPROBE can be beneficial to a variety of other distributed services and applications, which require efficient overlay network management.

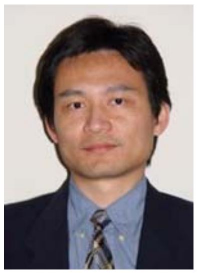
## Acknowledgements

Preliminary results of this study were presented in a paper at the 11th IEEE International Conference on Network Protocols. This work was conducted while C. Tang was a graduate student at Michigan State University. This work was supported in part by the U.S. Department of the Navy, Office of Naval Research under Grant No. N00014-01-1-0744. This work was also supported in part by U.S. National Science Foundation Grants CCR-9912407, EIA-0000433, EIA-0130724 and ITR-0313142.

## References

[1] Y. Chu, S. Rao, H. Zhang, A case for end system multicast, in: Proceedings of ACM Sigmetrics, June 2000, pp. 1–12.
[2] S. Banerjee, B. Bhattacharjee, C. Kommareddy, Scalable application layer multicast, in: Proceedings of ACM SIGCOMM, August 2002, pp. 205–220.
[3] A. Rowstron, P. Druschel, Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, in: Proceedings of 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), November 2001, pp. 329–350.
[4] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, A Scalable Content-Addressable Network, in: Proceedings of ACM SIGCOMM, August 2001, pp. 161–172.

[5] J. Li, REVERE: A fast event dissemination system, PhD Dissertation, UCLA, 2002.
[6] D. Andersen, H. Balakrishnan, F. Kaashoek, R. Morris, Resilient Overlay Networks, in: Proceedings of 18th ACM Symposium on Operating Systems Principles (SOSP), October 2001, pp. 131–145.
[7] A. Keromytis, V. Misra, D. Rubenstein, SOS: Secure Overlay Services, in: Proceedings of ACM SIGCOMM, August 2002, pp. 61–72.
[8] Performance measurement tools taxonomy, http://www.caida.org/tools/taxonomy/performance.xml.
[9] M. Faloutsos, P. Faloutsos, C. Faloutsos, On power-law relationships of the Internet topology, in: Proceedings of ACM SIGCOMM, September 1999, pp. 251–262.
[10] NLANR, National laboratory for applied network research, http://moat.nlanr.net/Routing/rawdata, 2000.
[11] P. K. McKinley, F. A. Samimi, J. K. Shapiro, C. Tang, Service Clouds: A distributed infrastructure for constructing autonomic communication services, in: Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC'06), (Indianapolis, Indiana), September 2006, pp. 341–348.
[12] A. Shaikh, M. Goyal, A. Greenberg, R. Rajan, K. Ramakrishnan, An OSPF topology server: Design and evaluation, IEEE Journal of Selected Areas in Communications: Special Issue on Recent Advances on Fundamentals of Network Management, vol. 20, May 2002, pp. 746–755.
[13] A. Bestavros, J. Byers, K. Harfoush, Inference and labeling of metric-induced network topologies, in: Proceedings of IEEE INFOCOM, June 2002, pp. 628–637.
[14] N. Spring, R. Mahajan, D. Wetherall, Measuring ISP topologies with Rocketfuel, in: Proceedings of ACM SIGCOMM, August 2002.
[15] M. Kwon, S. Fahmy, Topology-aware overlay networks for group communication, in: Proceedings of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2002), May 2002.
[16] W. Cui, I. Stoica, R. H. Katz, Backup path allocation based on a correlated link failure probability model in overlay networks, in: Proceedings of 10th IEEE International Conference on Network Protocols (ICNP'02), November 2002, pp. 236–247.
[17] C. Tang, Underlay Aware Overlay Networks. PhD thesis, Michigan State University, East Lansing, Michigan, USA, August 2005.
[18] R. Braynard, D. Kostic, A. Rodriguez, J. Chase, A. Vahdat, Opus: an overlay peer utility service, in: Proceedings of the 5th International Conference on Open Architectures and Network Programming (OPENARCH), June 2002.
[19] Y. Chen, D. Bindel, H. Song, R. Katz, An algebraic approach to practical and scalable overlay network monitoring, in: Proceedings of SIGCOMM, (Portland, Oregon), 2004.
[20] M. Jain, C. Dovrolis, End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput, in: Proceedings of ACM SIGCOMM, August 2002.
[21] Y. Bejerano, R. Rastogi, Robust monitoring of link delays and faults in IP networks, in: Proceedings of IEEE INFOCOM, April 2003.
[22] H. Ozmutlu, N. Gautam, R. Barton, Managing end-to-end network performance via optimized monitoring strategies, Journal of Network and Systems Management 10 (2002) 107–126.
[23] N. Duffield, F. Presti, Multicast inference of packet delay variance at interior network links, in: Proceedings of IEEE INFOCOM, March 2000, pp. 1351–1360.
[24] V. N. Padmanabhan, L. Qiu, H. J. Wang, Server-based inference of Internet link lossiness, in: Proceedings of IEEE INFOCOM, April 2003.
[25] N. Duffield, F. Presti, V. Paxson, D. Towsley, Inferring link loss using striped unicast probes, in: Proceedings of IEEE INFOCOM, April 2001, pp. 915–923.
[26] K. Harfoush, A. Bestavros, J. Byers, Measuring bottleneck bandwidth of targeted path segments, in: Proceedings of IEEE INFOCOM, April 2003.
[27] G. McLachlan, T. Krishnan, The EM Algorithm and Extensions, John Wiley & Sons, 1997.
[28] G. Liang, B. Yu, Pseudo likelihood estimation in network tomography, in: Proceedings of IEEE INFOCOM, April 2003.
[29] Y. Shavitt, X. Sun, A. Wool, B. Yener, Computing the unmeasured: an algebraic approach to Internet mapping, in: Proceedings of IEEE INFOCOM, pp. 1646–1654, April 2001.
[30] D. Johnson, Approximation algorithms for combinatorial problems, Journal of Computer and System Sciences 9 (1974) 256–278.
[31] C. Tang, P. K. McKinley, A distributed approach to topology-aware overlay path monitoring, in: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), March 2004.
[32] V. Chvatal, A greedy heuristic for the set-covering problem, Mathematics of Operations Research 4 (3) (1979) 233–235.
[33] E. Zegura, K. Calvert, S. Bhattacharjee, How to model an internetwork, in: Proceedings of IEEE INFOCOM, March 1996, pp. 40–52.
[34] J. Winick, S. Jamin, Inet 3.0: Internet topology generator, Tech. Rep. CSE-TR-456-02, Department of Computer Science, University of Michigan, Ann Arbor, Michigan, 2002.



**Chiping Tang** received both his Ph.D. and M.S. degree from Michigan State University, and a B.S. degree from Peking University, all in computer science. His research interests include Internet computing and mobile/wireless systems. He joined Microsoft Corporation in 2005. He may be contacted electronically at tangcp28@gmail.com.

**Philip K. McKinley** received the B.S. degree in mathematics and computer science from Iowa State University in 1982, the M.S. degree in computer science from Purdue University in 1983, and the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign in 1989. He is currently a Professor of Computer Science and Engineering at Michigan State University, where he has been on the faculty since 1990. He was previously a member of technical staff at Bell Laboratories in Naperville, Illinois. His current research interests include autonomic computing, digital evolution, and pervasive computing. He may be contacted electronically at mckinley@cse.msu.edu.