Queuing Network Topology Inference Using Passive Measurements

Yilei Lin, Ting He, and Guodong Pang
Pennsylvania State University, University Park, PA 16802, USA. Email: {yjl5282,tzh58,gup3}@psu.edu

***Abstract*—In this work, we revisit a classic problem of inferring a tree topology from end-to-end measurements originated by a single source, with two critical differences: (i) instead of relying on measurements with specific correlation across paths that often require active probing, we do not rely on any correlation and can thus utilize passive measurements; (ii) instead of inferring a logical topology that ignores certain nodes, we want to recover the physical topology. Our key idea is to utilize the detailed queuing dynamics inside the network to estimate the number of queues and a certain parameter (residual capacity) of each queue on each measurement path, and then use the estimated parameters as fingerprints to detect shared queues and infer the topology. To this end, we develop a Laplace-transform-based estimator to extract the parameters of a tandem of queues from end-to-end delays, and efficient algorithms to identify the parameters associated with the same queue and infer the topology accordingly. The inferred topology is guaranteed to converge to the ground truth, up to a permutation of queues traversed by the same paths, as the number of measurements increases. Our evaluations validate the proposed solutions against benchmarks and identify potential directions for further improvements.**

***Index Terms*—Network topology inference, passive measurement, queuing network, phase-type distribution.**

I. Introduction

Understanding the internal structure of a network, i.e., the *network topology*, is critical for a variety of tasks such as routing, content distribution, service placement, load balancing, and overlay construction. While network topology is traditionally maintained by the network administrator (e.g., via the help of local agents running Simple Network Management Protocol or OpenFlow) and discoverable by traceroute, obtaining this information for multi-domain networks such as the Internet is much more challenging due to the lack of internal support.

In these cases, *network tomography* provides a promising approach that infers the network topology from end-to-end measurements. Since introduction in the 1990s [1], a number of algorithms have been developed based on this idea, which use multicast measurements [2]–[4], their unicast-based approximations [5], [6], or network coding [7] to infer the routing tree rooted at each probing source, and further stitch the trees for multiple sources to form a general topology [8]. A common foundation of these works is a probing scheme that generates *specifically correlated measurements* across different paths, so that the correlation (caused by shared links/nodes) can be used to estimate the "lengths" of the shared portions of these paths. These shared path lengths can then be used to reveal the branching/joining points between different paths and thus the network topology.

Despite the extensive studies, existing topology inference algorithms have the limitations that (i) they mostly rely on *active probes* due to the specific type of correlation required by each algorithm, which increases the network load, and (ii) they can only infer a *logical topology* that ignores the degree-2 nodes between branching/joining points. In this work, we tackle these limitations by developing a topology inference algorithm that can utilize *passive measurements* with arbitrary (or no) correlation across paths, and recover the *physical topology* with possible degree-2 nodes. This is achieved by a fundamentally different approach that utilizes the detailed queuing dynamics inside the network. As a first step, we focus on inferring the tree topology rooted at a single source, and leave the extension to multi-source topologies to future work.

A. Related Work

Network topology inference: Our work belongs to a branch of network tomography aiming at inferring routing topologies using end-to-end measurements. The technique was originated based on the observation that correlated losses observed at multicast receivers can be used to infer the multicast tree [1], and was then extended to utilize a variety of multicast measurements, including losses [2], delays [3] and a combination of both [4]. As multicast is not widely supported, solutions based on unicast were proposed [5], [6]. These solutions, however, were based on specially-designed probing schemes such as stripes of back-to-back unicast packets [5] or "sandwiches" of small and large packets [6], both inducing correlated measurements at different receivers that can reveal performance metrics on the shared portions of end-to-end paths. While the above works aimed at inferring a tree topology rooted at a single source, later works (e.g., [8] and references therein) addressed more general topologies by merging trees rooted at multiple sources. However, these works are still based on multicast or its approximations, which requires active probing. Another line of works relies on network coding (e.g., [9] and references therein). These solutions rely on internal nodes that perform network coding, thus not applicable in current packet-switched networks.

In contrast, we take a fundamentally different approach of fingerprinting queues at internal nodes based on (possibly) uncorrelated end-to-end performance measurements, thus able to leverage passive measurements from existing data packets.

Queuing parameter inference: Our approach is based on the inference of queuing parameters from end-to-end measure-

This research was partly supported by the National Science Foundation under awards CNS-1946022 and CCF-1813219.

ments. To this end, a variety of parameters have been tackled in the context of communication networks. For example, it was shown in [10] that the difference between the delays measured when the buffer is full or empty can be used to estimate the buffer size at the bottleneck linkIn [11], packet arrival times and flow identifiers were used to detect bottleneck links shared between flows and estimate their bandwidths. In [12], periodic probes were used to detect the "dominant congested link" on a path and estimate the maximum queuing delay at this link. These works only focused on the bottleneck links, and while useful for performance diagnosis, did not provide sufficient information for topology inference.

In the context of generic queuing systems, the inference of queuing parameters has been posed as inverse problems, with several inversion techniques developed to infer input and service time characteristics from delay/loss measurements for a single queue [13]. However, when the system becomes more complex (e.g., a tandem of queues), inversion techniques became unstable [13], and solutions fell back to standard algorithms based on maximum likelihood estimation [13], [14]. We refer to [15] for a comprehensive bibliography in this space. To our knowledge, all the existing works assumed the queuing network topology to be known.

B. Summary of Contributions

We aim at inferring the topology of a queuing network modeling the connections from a given source to a given set of destinations, based on possibly uncorrelated measurements of the end-to-end delays. Our contributions are:

1) We propose a novel approach for topology inference that can utilize passive measurements and potentially recover the physical topology (of the queuing network) by exploiting the detailed queuing dynamics inside the network.

2) We develop a Laplace-transform-based estimator that can estimate the length and the residual capacities of a tandem of queues from end-to-end delays, which outperforms the maximum likelihood estimator (MLE) at finite sample sizes and is asymptotically consistent.

3) Based on the estimated queue parameters (i.e., residual capacities), we develop computationally efficient algorithms to identify the parameters associated with the same queue, and then construct a tree topology accordingly. The constructed topology is guaranteed to be identical to the ground truth, up to a permutation of queues traversed by the same set of measurement paths, if the parameter estimation is sufficiently accurate.

4) Our evaluations based on a real topology show that: (i) the proposed estimator outperforms MLE in both efficiency and accuracy in a wide range of settings, and (ii) for small topologies, the proposed topology inference algorithm can even outperform state-of-the-art solutions based on active probing due to its capability of discovering degree-2 nodes. Meanwhile, we find it difficult to accurately infer large topologies from uncorrelated unicast measurements, and identify potential directions for further improvements.

Roadmap. Section II formulates our problem, Section III addresses parameter estimation for a tandem of queues, Section IV addresses topology inference based on the estimated parameters, Section V evaluates the performance of the proposed algorithms, and Section VI concludes the paper.

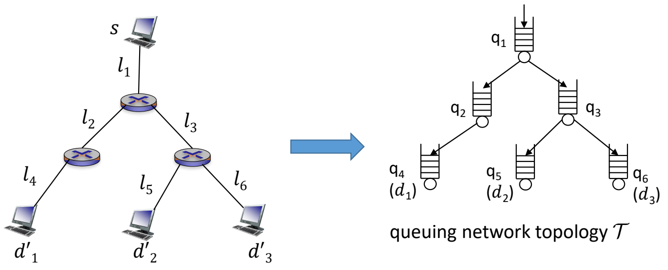


Fig. 1. Queuing network model.

II. Problem Formulation

A. Network Model

Given the routing tree $\mathcal{T}'$ connecting a given source s to a given set of destinations $\{d'_1, \ldots, d'_N\}$, we model this topology by a queuing network as shown in Fig. 1, where each queue q_i models the outgoing interface of a link $l_i \in \mathcal{T}'$. This model is motivated by the fact that queuing in packet-switched networks typically occurs at the outgoing interfaces. It is easy to see that the resulting queuing network has the topology of a rooted tree, denoted by $\mathcal{T}$, where each vertex represents a queue that corresponds to a link in the original routing topology. One can easily obtain the original topology from $\mathcal{T}$. Let $d_i \in \mathcal{T}$ $(i = 1, \ldots, N)$ denote the leaf modeling the access link for destination d'_i, and p_i denote the path from the root of $\mathcal{T}$ to d_i.

We model each q_i as an $M/M/1$ queue, where the sojourn time models the delay imposed by link l_i on a packet traversing l_i. Specifically, let λ_i denote the unknown load on link l_i and μ_i denote the unknown capacity of this link, both measured in packets per second. We assume that $\mu_i > \lambda_i$, which guarantees queue stability. Then it is well-known [16] that the sojourn time T_i of q_i in the steady state is exponentially distributed with parameter $\delta_i := \mu_i - \lambda_i$ that represents the *residual capacity*, i.e., the PDF of T_i is $w(t_i) = \delta_i e^{-\delta_i t_i}$ $(t_i > 0)$. Moreover, we assume that the sojourn times of a given packet at different queues are independent of each other in the steady state, and hence the end-to-end delay on path p_j follows the *hypoexponential distribution* with parameters $(\delta_i)_{q_i \in p_j}$, where "$q_i \in p_j$" means for each queue q_i on path p_j.

Remark: Our assumptions are automatically satisfied if $\mathcal{T}$ is a Jackson network with $M/M/1$ queues, which is a commonly-used model in queuing theory. We refer to [13, Section 6.2] for a detailed discussion on the realism of this model. We further note that we *do not* assume the same packet to have independent service times at different queues (these times will be correlated); what we assume is that different queues receive largely disjoint cross-traffic, and thus a measurement packet will incur largely independent waiting times at different queues and hence largely independent sojourn times (assumed to be dominated by the waiting times).

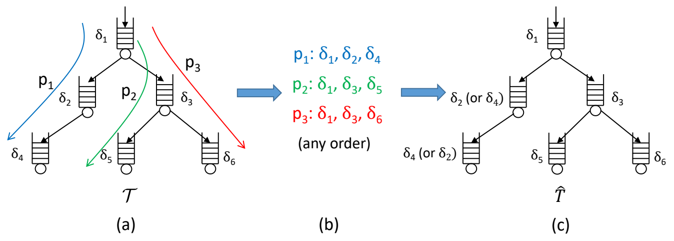


Fig. 2. Motivating example.

B. Objective

Given i.i.d. measurements $\boldsymbol{x}_j := (x_{j,h})_{h=1}^n$ of the end-to-end delay on each path p_j ($x_{j,h}$: the h-th measurement on path p_j), we want to infer the queuing network topology $\mathcal{T}$. These measurements can be obtained from active probes, passive monitoring of data packets, or a combination of both. In contrast to the previous works [5], [6], *we do not rely on specific correlation of measurements across paths*, and thus can utilize passive measurements that can be arbitrarily correlated (possibly uncorrelated) across paths. The temporal independence assumption for measurements on the same path can be justified by ensuring sufficient spacing between measurements as in [6]. For the problem to be well-defined, we assume that the link loads remain fixed during measurement, which can be achieved by collecting the measurements during off-peak hours so that there is minimal fluctuation in traffic.

C. Motivating Example

We illustrate why the problem is solvable by a simple example. As illustrated in Fig. 2 (a), $\mathcal{T}$ models the connection from a source to three destinations via paths p_1, p_2, and p_3. The end-to-end delay on p_j ($j = 1, \ldots, 3$) follows a hypoexponential distribution with the parameters listed in Fig. 2 (b), which can be accurately estimated after collecting sufficiently many measurements. Under the assumption that different queues have different parameters, we can infer that there is a queue shared by all three paths with parameter δ_1 and another queue shared by paths p_2 and p_3 with parameter δ_3. Due to the tree structure, we can then infer the topology as in Fig. 2 (c), where the only uncertainty is in the order of the queues with parameters δ_2 and δ_4 on p_1. Since this order does not affect the end-to-end delay distribution on any path, Fig. 2 (c) is the most accurate estimate that can be obtained.

Following the idea in this example, we will develop our solution in two steps: (1) estimating the δ-parameters (i.e., residual capacities) for a tandem of queues from their end-to-end delays, and then (2) inferring the queuing network topology by merging the queues with sufficiently similar δ-parameters.

III. Parameter Estimation for Tandem Queues

We first focus on estimating the δ-parameters for a tandem of queues modeling a given path. For simplicity, we will omit the path index and simply denote the delay measurements on this path by $\boldsymbol{x} := (x_h)_{h=1}^n$ and the queue parameters by $\boldsymbol{\delta} := (\delta_1, \ldots, \delta_K)$, where K is an upper bound on the number of queues per path (i.e., the maximum hop count between the source and the destinations in the routing topology). As the actual number of queues on a given path is unknown, we use this upper bound to define the number of unknown parameters. As the order of queues does not affect the end-to-end delay distribution and hence cannot be identified from the delay measurements from a single path (later in Section IV we will show how to identify their order up to segments in the tree), we assume $\delta_1 \leq \cdots \leq \delta_K$ when comparing the estimated and the true parameters. If the actual number of queues is K^* ($K^* < K$), then the last $K - K^*$ queues are *virtual queues* with no additional delay, i.e., $\delta_{K^*+1} = \cdots = \delta_K = \infty$. Our estimation algorithm (Algorithm 1) will also estimate K^*.

A. Maximum Likelihood Estimation (MLE)

Assuming that $K = K^*$, [13] proposed to apply the MLE to this problem. More generally, MLE is considered the state-of-the-art estimator for phase-type distributions [17], which includes the hypoexponential distribution as a special case. Specifically, assuming that $\delta_i \neq \delta_j$ for any $i \neq j$, we can express the PDF of the end-to-end delay as:

$$g(x;\, \boldsymbol{\delta}) = \sum_{i=1}^{K} \delta_i e^{-x\delta_i} \left(\prod_{j=1, j\neq i}^{K} \frac{\delta_j}{\delta_j - \delta_i} \right), \tag{1}$$

and the MLE aims at finding the value of $\boldsymbol{\delta}$ that maximizes the log-likelihood $\sum_{h=1}^n \log g(x_h;\, \boldsymbol{\delta})$. If solved exactly, the MLE has a desirable property that it is asymptotically efficient under regularity conditions[1], i.e., as the number of measurements increases, it converges to the true parameter at a rate approximating the *Cramér-Rao bound* [18].

However, as the log-likelihood function is non-concave, computing the MLE is challenging. To address this challenge, various algorithms have been proposed to compute an approximation to MLE [17]. In particular, the *Expectation-Maximization (EM)* algorithm is guaranteed to converge to a local maximum and was adopted to solve this problem in [13]. However, we find EM to be extremely slow in our case due to the calculation of numerical integration, which combined with its known slow convergence [14] makes it impractical for our problem.

B. Estimation based on Laplace Transform

Motivated by the need to improve the estimation speed and accuracy, we exploit estimators based on the Laplace transform. Defined as $\mathbf{E}[e^{-sX}]$ for a random variable X, the Laplace transform uniquely determines the distribution of X (except on a set of Lebesgue measure zero), and can be numerically inverted to compute the CDF/PDF of the distribution [19]. While the transform has been used to estimate PDF/CDF from data [20], [21], to our knowledge, we are the first to apply it to parameter estimation.

The Laplace transform appears promising for our problem due to its property that if X is a summation of independent random variables $X_1, \ldots, X_K$, then $\mathbf{E}[e^{-sX}] = \prod_{i=1}^K \mathbf{E}[e^{-sX_i}]$, thus providing a simple target function for

[1]These conditions are that (i) the log-likelihood function is twice differentiable, and (ii) the Fisher Information Matrix is non-zero, both satisfied in our case.

fitting. Specifically, since the Laplace transform of an exponential random variable with parameter δ_i is $\delta_i/(\delta_i+s)$ $(s > -\delta_i)$, the Laplace transform of the end-to-end delay is

$$L(s;\, \boldsymbol{\delta}) := \prod_{i=1}^{K} \frac{\delta_i}{\delta_i + s}, \quad s > -\min_{i=1,\ldots,K} \delta_i. \tag{2}$$

Our idea is to estimate the Laplace transform at a predetermined set of values for s, and find the parameter $\boldsymbol{\delta}$ yielding the best fit.

Estimator: By definition, the empirical Laplace transform

$$\hat{L}(s;\, \boldsymbol{x}) := \frac{1}{n}\sum_{h=1}^{n} e^{-s x_h} \tag{3}$$

gives an unbiased estimate of $L(s;\, \boldsymbol{\delta})$. Given the empirical Laplace transform, we propose to estimate $\boldsymbol{\delta}$ by solving

$$\min \quad \sum_{s\in S} |L(s;\, \boldsymbol{\delta}) - \hat{L}(s;\, \boldsymbol{x})| \tag{4a}$$

$$\text{s.t. } 0 < \delta_1 \le \cdots \le \delta_K, \tag{4b}$$

i.e., fitting the empirical Laplace transform at $s \in S$ by minimizing the absolute error, where S is a design parameter.

We note that there are other ways to use the Laplace transform for estimating $\boldsymbol{\delta}$. For example, we can rewrite the Laplace transform (2) as

$$L(s;\, \boldsymbol{\delta}) = \frac{a_0}{a_0 + a_1 s + \cdots a_{K-1}s^{K-1} + s^K}, \tag{5}$$

where the coefficients are related to $\boldsymbol{\delta}$ by

$$a_j := \sum_{A\subseteq\{1,\ldots,K\}, |A|=j} \ \prod_{i\in\{1,\ldots,K\}\setminus A} \delta_i, \ \ j = 0,\ldots,K-1. \tag{6}$$

Then we can estimate the coefficients $a_0,\ldots,a_{K-1}$ from the empirical moments or the empirical Laplace transform as in [20], and solve the system of equations (6) for $\boldsymbol{\delta}$. We can also use other error measures in (4a) (e.g., squared error). However, we find that the proposed estimator achieves the best performance (see Fig. 7–8).

Performance: The proposed estimator is *consistent* under sufficiently large $|S|$ in the following sense.

Theorem III.1. As $n \to \infty$, (4) has a unique optimal solution that equals the ground truth if $|S| > K$.

Proof. As $n \to \infty$, $\hat{L}(s;\, \boldsymbol{x})$ will converge almost surely to $L(s;\, \boldsymbol{\delta}^*)$, where $\boldsymbol{\delta}^*$ denotes the true parameter. Suppose that $\exists \hat{\boldsymbol{\delta}} \neq \boldsymbol{\delta}^*$ that is optimal for (4) at convergence. It must satisfy

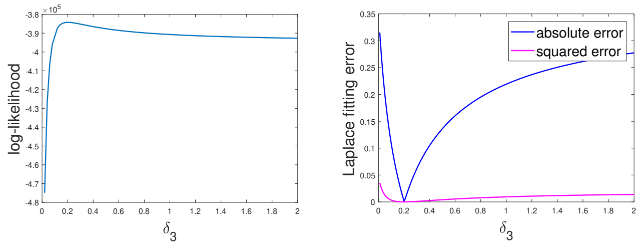
$$\sum_{s\in S} |L(s;\, \hat{\boldsymbol{\delta}}) - L(s;\, \boldsymbol{\delta}^*)| \le 0, \tag{7}$$

and thus

$$\prod_{i=1}^{K} \frac{\hat{\delta}_i}{\hat{\delta}_i + s} - \prod_{i=1}^{K} \frac{\delta_i^*}{\delta_i^* + s} = 0, \quad \forall s \in S. \tag{8}$$

The left-hand side of (8) can be converted to an order-K polynomial of s, which has at most K distinct solutions. This contradicts with the assumption of $|S| > K$, thus completing the proof. □

At finite sample sizes, we find it easier to find a good solution based on the proposed objective in (4a) than the MLE objective (see Fig. 7–8). Intuitively, this is because our objective function has a larger gradient norm around the optimal value, as illustrated in Fig. 3, and hence optimization algorithms are less likely to stop at suboptimal values. Fig. 3 (b) further justifies our choice of the absolute error in (4a) as opposed to the commonly-used squared error. Similar observations have been obtained under other settings.



(a) MLE (b) Laplace fitting

Fig. 3. Comparison of objective functions (true parameter $\boldsymbol{\delta}^* = (0.1, 0.15, 0.2, \infty)$, $n = 10^5$, $S = \{0, 0.1, 0.2, \ldots, 1\}$, $\delta_i = \delta_i^*$ for $i \neq 3$).

Design of S: Besides the condition in Theorem III.1, the values in S also affect the accuracy of the proposed estimator at finite samples sizes. Intuitively, S should contain a diverse range of values to provide a good description of the Laplace transform. It has been suggested in [20] that these values should be evenly distributed. We also find that increasing the density of points in S helps (see Fig. 6 (a)), at the cost of increased complexity.

Algorithm: Although in theory virtual queues with $\delta_i = \infty$ can be used to accommodate overestimation of the number of queues, in practice such overestimation tends to increase the error in estimating $\boldsymbol{\delta}$ as optimization algorithms used to solve (4) will always give finite values. Therefore, we propose to test the fitting error under various numbers of queues up to K and select the estimate with the minimum fitting error, as shown in Algorithm 1. Lines 6–7 are used to find a good initial value of $\boldsymbol{\delta}$ by iteratively optimizing one δ_i at a time, while fixing the other δ_j for $j \neq i$. We find that optimization based on this initialization outperforms optimization from an arbitrary initial guess (see Fig. 6 (b)).

IV. Queuing Network Topology Inference

Based on the inferred δ-parameters from each tandem of queues for which the end-to-end delays can be measured, i.e., from each path p_i $(i = 1,\ldots,N)$, we are now ready to infer the queuing network topology $\mathcal{T}$. The key observation is that if each queue has a distinct δ-parameter, which is likely to be satisfied in production networks due to the unique mix of cross-traffic on each link, then we can use this parameter as a "fingerprint" of the queue to detect the queues shared by different paths.

Specifically, given the inferred parameters $\boldsymbol{\delta}_i := (\delta_{ij})_{j=1}^{k_i}$ for path p_i $(i = 1,\ldots,N)$, we view p_i as a tandem of k_i queues, denoted by $\boldsymbol{q}_i := (q_{ij})_{j=1}^{k_i}$. We will develop efficient algorithms to merge these N tandems of queues into a tree-shaped queuing network in two steps: (1) inferring which input queues represent the same queue in the underlying queuing network, and (2) inferring the queuing network topology accordingly. We now tackle these two steps separately.

Algorithm 1: Laplace-based Tandem Queue Inference

```
input : end-to-end delays $\boldsymbol{x} := (x_h)_{h=1}^n$, maximum
        number of queues $K$, input parameters for
        Laplace transform $S$
output: estimated queue parameters $\hat{\boldsymbol{\delta}}$
1  initialization: $e_{\min} \leftarrow \infty$, $\hat{\boldsymbol{\delta}} \leftarrow \emptyset$;
2  for $s \in S$ do
3      $\hat{L}(s;\boldsymbol{x}) \leftarrow \frac{1}{n}\sum_{h=1}^n e^{-sx_h}$;
4  for $k = 1$ to $K$ do
5      let $\boldsymbol{\delta}^{(0)}$ be an arbitrary initial guess with length $k$;
6      for $i = 1$ to $k$ do
7          $\delta_i^{(0)} \leftarrow \mathrm{argmin}_{\delta_i} \sum_{s\in S} |L(s;\boldsymbol{\delta}) - \hat{L}(s;\boldsymbol{x})|$,
            where $\delta_j = \delta_j^{(0)}$ for all $j \neq i$;
8      $\boldsymbol{\delta}^{(1)} \leftarrow \mathrm{argmin}_{\boldsymbol{\delta}} \sum_{s\in S} |L(s;\boldsymbol{\delta}) - \hat{L}(s;\boldsymbol{x})|$ with
        initial value $\boldsymbol{\delta} = \boldsymbol{\delta}^{(0)}$;
9      if $e := \sum_{s\in S} |L(s;\boldsymbol{\delta}^{(1)}) - \hat{L}(s;\boldsymbol{x})| < e_{\min}$ then
10         $e_{\min} \leftarrow e$ and $\hat{\boldsymbol{\delta}} \leftarrow \boldsymbol{\delta}^{(1)}$;
```

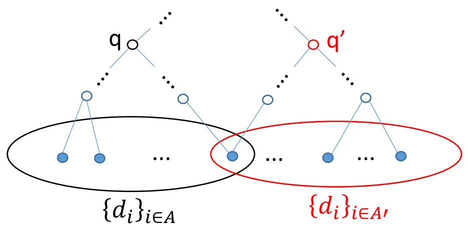


Fig. 4. A queue q of category A and a queue q' of category A' cannot coexist in $\mathcal{T}$ if $A \cap A' \neq \emptyset$, $A \not\subseteq A'$, and $A' \not\subseteq A$.

A. Inferring Associated Queues

We define a set of input queues, each modeling a link traversed by a different path, as a set of *associated queues*, if they model the same link in the routing topology (and hence correspond to the same queue in the queuing network $\mathcal{T}$). Algorithm 2 shows our algorithm to identify the sets of associated queues based on similarities of the estimated δ-parameters.

1) Algorithm: Specifically, define

$$D_{\{q_{i_1j_1},\ldots,q_{i_kj_k}\}} := \max\{\delta_{i_1j_1},\ldots,\delta_{i_kj_k}\} - \min\{\delta_{i_1j_1},\ldots,\delta_{i_kj_k}\} \quad (9)$$

as the error in associating the input queues $\{q_{i_1j_1},\ldots,q_{i_kj_k}\}$ to the same queue in $\mathcal{T}$. We see that if the maximum estimation error for any δ_{ij} is $\Delta/2$, then a set of input queues $\{q_{i_1j_1},\ldots,q_{i_kj_k}\}$ may represent the same queue in $\mathcal{T}$ only if $D_{\{\delta_{i_1j_1},\ldots,\delta_{i_kj_k}\}} \leq \Delta$. Hereafter, we refer to a set s of input queues (each on a different path) satisfying $D_s \leq \Delta$ as a *candidate set*, meaning a candidate set of associated queues, where Δ is a design parameter that controls the tradeoff between detecting truly associated queues and not detecting non-associated queues as associated queues.

However, the feasibility of different candidate sets (even if they are disjoint) cannot be determined independently due to the constraint of tree topology. To see this, we define the *category* of a set of input queues $s := \{q_{i_1j_1},\ldots,q_{i_kj_k}\}$ as the set of indices of the paths traversing the queues in s, denoted by

$$c(s) := \{i_1,\ldots,i_k\}. \quad (10)$$

We will also refer to the indices of the paths traversing a queue q in $\mathcal{T}$ as the category of q. Clearly, if the queues in s represent the same queue q in $\mathcal{T}$, then s and q have the same category. For example, the inference results in Fig. 2 (b) suggest that there is a queue with parameter δ_1 that has category $\{1,2,3\}$. The problem is that if we associate a set of input queues of category $A \subseteq \{1,\ldots,N\}$ with the same queue q in $\mathcal{T}$, then q will reside on the tree branch containing "destinations" $\{d_i\}_{i\in A}$ (recall that d_i is the queue modeling the access link of destination d'_i), and thus there cannot be another queue q' of category A' for any A' satisfying

$$A' \cap A \neq \emptyset,\ A' \not\subseteq A,\ \text{and } A \not\subseteq A', \quad (11)$$

as illustrated in Fig. 4. We say that two candidate sets s and s' *conflict with each other* if $c(s)$ and $c(s')$ satisfy (11).

To avoid such conflict, our idea is to iteratively select candidate sets via a greedy procedure, where each iteration selects the candidate set not conflicting with the existing selections that has the minimum error defined as in (9). However, a straightforward implementation of this idea will incur an exponential complexity as there are $O(K^N)$ candidate sets.

Algorithm 2 avoids the exponential complexity by only searching among the candidate sets that may achieve the minimum error. Specifically, let Q denote the currently selected candidate sets and $\tilde{Q}$ the candidate sets that will be searched in the next iteration. For each $s \in \tilde{Q}$, let $F_s \in \{0,1\}$ indicate whether the candidate set s is feasible, i.e., not conflicting with any $s' \in Q$. Starting by selecting all the singletons $\{q_{ij}\}$ into Q as $D_{\{q_{ij}\}} = 0$ (lines 2–3), we see that: (i) for the first iteration, the minimum error among the candidate sets outside Q must be achieved at a set of two queues; (ii) for each of the subsequent iterations, the minimum error among the feasible candidate sets outside Q must be achieved at the union of two sets in Q. This allows us to initialize (lines 4–8) and update (lines 13–29) $\tilde{Q}$ and its corresponding properties only for the candidate sets that may be selected in the next iteration. The algorithm continues until all the candidate sets not conflicting with the already-selected sets have been considered (line 9).

2) Complexity: Recall that K is the maximum number of queues per path. By design, $|Q|$ starts at $O(KN)$ and reduces by one in each iteration (as two sets in Q will be replaced by their union). Moreover, $\tilde{Q}$ contains the union of each pair of sets in Q, and hence $|\tilde{Q}| = O(K^2N^2)$. This implies a space complexity of $O(K^2N^3)$, dominated by the space for storing $\tilde{Q}$, as the cardinality of each set in $\tilde{Q}$ is at most N. For time complexity, the initialization (lines 2–8) takes $O(K^2N^2)$, each while loop (lines 10–29) takes $O(K^3N^4)$, dominated by the update of $(F_s)_{s\in\tilde{Q}}$ in lines 14–19 (as the conflict between two sets can be checked in $O(N)$), and the while loop is repeated $O(KN)$ times (each reducing $|Q|$ by one). The total time complexity is thus $O(K^4N^5)$. Note that this is only the worst-case complexity when all the considered sets are candidate sets; in practice, the complexity will be lower with a smaller Δ.

3) Correctness: We show that the inference by Algorithm 2 will be accurate if the input parameters are sufficiently accurate. For each queue $e \in \mathcal{T}$, let δ_e^* denote its true parameter,

Algorithm 2: Inference of Associated Queues

```
input : inferred parameters $\boldsymbol{\delta}_1, \cdots, \boldsymbol{\delta}_N$ for all the
        paths ($\boldsymbol{\delta}_i = (\delta_{ij})_{j=1}^{k_i}$); threshold $\Delta$
output: Collection $Q$ of sets of queues, where each
        $s \in Q$ is a set of input queues inferred to be
        associated with the same queue in $\mathcal{T}$
1  $Q \leftarrow \emptyset$; $\tilde{Q} \leftarrow \emptyset$;
2  for $i = 1, \ldots, N$ do
3      $Q \leftarrow Q \cup \{\{q_{i1}\}, \ldots, \{q_{ik_i}\}\}$;
4  for $\{q_{i_1j_1}\}, \{q_{i_2j_2}\} \in Q$ with $i_1 \neq i_2$ do
5      if $|\delta_{i_1j_1} - \delta_{i_2j_2}| \leq \Delta$ then
6          $\tilde{Q} \leftarrow \tilde{Q} \cup \{\{q_{i_1j_1}, q_{i_2j_2}\}\}$;
7          $D_{\{q_{i_1j_1}, q_{i_2j_2}\}} \leftarrow |\delta_{i_1j_1} - \delta_{i_2j_2}|$;
8          $F_{\{q_{i_1j_1}, q_{i_2j_2}\}} \leftarrow 1$;
9  while $\exists s \in \tilde{Q}$ with $F_s = 1$ do
10     $s^* \leftarrow \mathrm{argmin}_{s \in \tilde{Q}, F_s = 1} D_s$;
11     $Q \leftarrow Q \cup \{s^*\}$;
12     $Q \leftarrow Q \setminus \{s \in Q : s \subset s^*\}$;
13     $\tilde{Q} \leftarrow \tilde{Q} \setminus \{s \in \tilde{Q} : s \cap s^* \neq \emptyset\}$;
14     for $s' \in \tilde{Q}$ do
15         $F_{s'} \leftarrow 1$;
16         for $s \in Q$ do
17             if $s'$ conflicts with $s$ then
18                 $F_{s'} \leftarrow 0$;
19                 break;
20     for $s \in Q$ such that $c(s) \cap c(s^*) = \emptyset$ do
21         $d \leftarrow \max_{q_{ij} \in s \cup s^*} \delta_{ij} - \min_{q_{ij} \in s \cup s^*} \delta_{ij}$;
22         if $d \leq \Delta$ then
23             $\tilde{Q} \leftarrow \tilde{Q} \cup \{s \cup s^*\}$;
24             $D_{s \cup s^*} \leftarrow d$;
25             $F_{s \cup s^*} \leftarrow 1$;
26             for $s' \in Q$ do
27                 if $s \cup s^*$ conflicts with $s'$ then
28                     $F_{s \cup s^*} \leftarrow 0$;
29                     break;
```

and s_e the true set of all the input queues associated with e.

Theorem IV.1. Let $\Delta^* := \min_{e,e' \in \mathcal{T}, e \neq e'} |\delta_e^* - \delta_{e'}^*|$. Algorithm 2 will output $Q = \{s_e\}_{e \in \mathcal{T}}$ if every input parameter δ_{ij} is associated with some $e \in \mathcal{T}$ and $|\delta_{ij} - \delta_e^*| \leq \frac{1}{2}\Delta < \frac{1}{4}\Delta^*$.

Proof. As the estimation error for any δ_{ij} is no more than $\Delta/2$, we must have $\delta_{ij} \in [\delta_e^* - \Delta/2,\ \delta_e^* + \Delta/2]$ for each $q_{ij} \in s_e$. Thus, $D_{s_e} \leq \Delta$, making s_e and its subsets candidate sets. Meanwhile, for any $e' \in \mathcal{T}$ such that $e' \neq e$, we will have $|\delta_{e'}^* - \delta_e^*| \geq \Delta^* > 2\Delta$. Hence, the estimated parameter $\delta_{i'j'}$ of any input queue associated with e' must satisfy $|\delta_{i'j'} - \delta_e^*| > 3\Delta/2$, and thus $|\delta_{i'j'} - \delta_{ij}| > \Delta$ for all $q_{ij} \in s_e$. This means that a set of input queues s will be considered a candidate set by Algorithm 2 if and only if $s \subseteq s_e$ for some $e \in \mathcal{T}$.

Moreover, we claim that the final output Q of Algorithm 2 must contain s_e for each $e \in \mathcal{T}$. We prove this by a top-down induction. First, the set s_{e_0} for e_0 at the root of $\mathcal{T}$ will not conflict with any other set and thus must be selected into Q. Second, if for a given $e \in \mathcal{T}$, $s_{e'} \in Q$ for each ancestor e' of e, then s_e will not conflict with any $s \in Q$ (associated with some $e'' \in \mathcal{T}$), since $c(s) \supseteq c(s_e)$ if e'' is an ancestor of e, $c(s) \cap c(s_e) = \emptyset$ if e'' is on a different branch from e, and $c(s) \subseteq c(s_e)$ if e'' is at or below e on the same branch, and thus s_e must be selected into Q. The proof completes by noting that no $s \subset s_e$ for any $e \in \mathcal{T}$ will be in Q due to line 12. □

B. Constructing Queuing Network Topology

The result of Algorithm 2 helps to infer the queuing network $\mathcal{T}$ by revealing the set of queues and the position of each queue. Specifically, if Algorithm 2 infers that a set of input queues $\{q_{i_1j_1}, \ldots, q_{i_kj_k}\}$ correspond to the same queue q in $\mathcal{T}$, then q must have category $\{i_1, \ldots, i_k\}$, and hence reside on the tree branch containing "destinations" $\{d_{i_1}, \ldots, d_{i_k}\}$. We now use this idea to construct the tree.

1) Algorithm: Algorithm 3 constructs the tree by going through the sets of associated queues inferred by Algorithm 2 in the increasing order of cardinality (line 2), breaking ties arbitrarily, and constructing a vertex to represent each set (line 3). This results in a bottom-up approach that constructs the tree from the leaves to the root. At any time, the set R contains the top-most vertex in each constructed branch. After constructing a new vertex v_s, the algorithm will connect it with each vertex $v_{s'} \in R$ that is on the same branch as v_s (indicated by $c(s') \cap c(s) \neq \emptyset$) and update R (lines 4–8). If Q does not contain any set of category $\{1, \ldots, N\}$, then the constructed topology will be a forest, in which case we merge the roots to form a tree (line 10).

Algorithm 3: Topology Construction

```
input : inferred parameters $\boldsymbol{\delta}_1, \cdots, \boldsymbol{\delta}_N$, output $Q$ of
        Algorithm 2
output: inferred topology $\widehat{\mathcal{T}}$
1  $\widehat{\mathcal{T}} \leftarrow \emptyset$; $R \leftarrow \emptyset$;
2  for $s \in Q$ in increasing order of $|s|$ do
3      create a vertex $v_s$ in $\widehat{\mathcal{T}}$ with parameter
         $\delta_{v_s} = \mathrm{mean}(\{\delta_{ij}\}_{q_{ij} \in s})$;
4      for $v_{s'} \in R$ do
5          if $c(s') \cap c(s) \neq \emptyset$ then
6              create an edge $(v_s, v_{s'})$ in $\widehat{\mathcal{T}}$;
7              $R \leftarrow R \setminus \{v_{s'}\}$;
8      $R \leftarrow R \cup \{v_s\}$;
9  if $|R| > 1$ then
10     merge all the vertices in $R$;
```

2) Complexity: Recall from Section IV-A2 that $|Q| = O(KN)$. Moreover, $|R| = O(N)$, as after constructing the N leaves, each new vertex will replace at least one existing vertex in R. This implies a space complexity of $O(KN^2)$, dominated by the space for storing Q, and a time complexity of $O(KN^3)$, dominated by line 5 (as there are $O(KN)$ loops in line 2, $O(N)$ loops in line 4, and $|c(s)| \leq N$ for all $s \in Q$).

3) Correctness: We say that v is a *branching point* in $\mathcal{T}$ if it is a vertex with at least two children. We say that two vertices v_1, v_2 in $\mathcal{T}$ are *on the same segment* if they are traversed by the same set of root-to-leaf paths. Based on

Algorithm 4: Queuing Network Topology Inference

```
input : paths $\{p_i\}_{i=1}^N$, #measurements per path $n$,
        maximum #queues per path $K$, parameter for
        Laplace transform $S$, threshold $\Delta$
output: inferred topology $\widehat{\mathcal{T}}$
1 for $i = 1, \ldots, N$ do
2     measure the delays $\boldsymbol{x}_i := (x_{i,h})_{h=1}^n$ of $p_i$;
3     $\hat{\boldsymbol{\delta}}_i \leftarrow$ parameters for $p_i$ inferred by Algorithm 1 or
      MLE based on $\boldsymbol{x}_i$, $K$, and $S$ (if applicable);
4 $Q \leftarrow$ sets of associated queues inferred by
  Algorithm 2 based on $(\hat{\boldsymbol{\delta}}_i)_{i=1}^N$ and $\Delta$;
5 $\widehat{\mathcal{T}} \leftarrow$ topology constructed by Algorithm 3 based on
  $(\hat{\boldsymbol{\delta}}_i)_{i=1}^N$ and $Q$;
```

these concepts, we will show that the output of Algorithm 3 is correct if its input Q is correct in the following sense.

Theorem IV.2. If $Q = \{s_e\}_{e\in\mathcal{T}}$, where s_e is the set of all the input queues associated with queue e, then the topology $\widehat{\mathcal{T}}$ constructed by Algorithm 3 will be identical to $\mathcal{T}$, except that vertices on the same segment may be permuted.

Proof. For each vertex e in $\mathcal{T}$, let $\tilde{e}$ denote the first branching point at or below e, and v_{s_e} the vertex in $\widehat{\mathcal{T}}$ corresponding to e. As each path traversing a vertex e' in $\mathcal{T}$ must traverse its parent e, we have $c(s_{e'}) \subseteq c(s_e)$, and $c(s_{e'}) \subset c(s_e)$ if e is a branching point. Thus, when Algorithm 3 creates a vertex v_{s_e} based on $s_e \in Q$, the vertices $\{v_{s_{e'}}\}_{e'\in E'}$ for each child e' of $\tilde{e}$ must have been created and satisfy $c(s_{e'}) \cap c(s_e) \neq \emptyset$, and every $v_{s_{e''}}$ for e'' on a different branch from e must have $c(s_{e''}) \cap c(s_e) = \emptyset$. If s_e is the first considered set of category $c(s_e)$, then v_{s_e} will be connected to $v_{s_{e'}}$ ($\forall e' \in E'$); otherwise, v_{s_e} will be connected to the top-most vertex $v_{s_{e_0}}$ with $c(s_{e_0}) = c(s_e)$ (since $v_{s_{e_0}}$ will have replaced $\{v_{s_{e'}}\}_{e'\in E'}$ in R), where e_0 and e must be on the same segment in $\mathcal{T}$. Vertices on the same segment will tie in line 2 and thus may be created in any order. □

Remark: The order of vertices (each representing a queue) on the same segment of $\mathcal{T}$ does not affect the end-to-end delay distribution on any path and is hence not identifiable.

C. Summary of Solution

Algorithm 4 summarizes our overall solution. Theorems III.1, IV.1, and IV.2 together guarantee that the inferred topology will converge to the ground truth (up to a permutation of queues on the same segment) as the number of measurements per path increases.

V. Performance Evaluation

We evaluate the proposed algorithms against benchmarks based on real Internet topologies.

A. Simulation Setup

1) Topology Generation: We generate the ground-truth tree topologies based on a Rocketfuel *Autonomous System (AS)* topology [22], which represents IP-level connections between the routers in AS6461 of Abovenet, containing 182 nodes and 294 links. Given a maximum path length (measured in #nodes) of K, we start from a randomly selected high-degree node s' (with degree > 6) as the "source" and perform a breadth first search (BFS) to obtain a tree of height K. We then randomly pick N of the leaves of this tree as destinations, forming a subtree with N leaves, which is used as the ground truth $\mathcal{T}$ for topology inference.

The above topology construction guarantees that the number of nodes (each representing a queue) on each root-to-leaf path is bounded by K. We randomly assign each node q_i a parameter $\delta_i \in (0, 2)$, while making sure that $|\delta_i - \delta_j| \geq \Delta^*$ for $i \neq j$, where Δ^* denotes the minimum (possible) difference between the parameters of different queues ($\Delta^* = 0$ means they can be arbitrarily close).

2) Benchmarks: For estimating the δ-parameters from the end-to-end delays on each path, we compare the proposed estimator (Algorithm 1) with the MLE (solved by the Nelder-Mead simplex method [23]), and other Laplace-based estimators discussed in Section III-B, including the two methods proposed by Harris et al. [20] ('Harris 1', 'Harris 2'), and a variation of the proposed estimator that replaces the absolute error (4a) by the squared error ('squared').

For topology inference, existing solutions relied on correlated measurements provided by multicast or its approximations (see Section I-A), and are thus not applicable to our measurements. Nevertheless, as these solutions aim at inferring the *multicast tree*, which is a logical topology obtained by merging nodes on the same segment (e.g., Fig. 5 (c)), we will compare with the accuracy of approximating the ground-truth physical topology by the corresponding multicast tree, which can be estimated by *Rooted Neighbor Joining* (RNJ) [5] from sufficiently many multicast measurements. The multicast tree represents the asymptotic accuracy of existing solutions that employ active probing, and thus should be treated as a "performance upper bound".

3) Metrics: We evaluate the accuracy of parameter estimation by the relative error, defined as $\|\hat{\boldsymbol{\delta}} - \boldsymbol{\delta}^*\|_1/\|\boldsymbol{\delta}^*\|_1$, where $\hat{\boldsymbol{\delta}}$ is the estimate and $\boldsymbol{\delta}^*$ the ground truth. If their dimensions $\dim(\hat{\boldsymbol{\delta}})$ and $\dim(\boldsymbol{\delta}^*)$ are different, we fill $\hat{\boldsymbol{\delta}}$ with the maximum value if $\dim(\hat{\boldsymbol{\delta}}) < \dim(\boldsymbol{\delta}^*)$, and discard the largest $\dim(\hat{\boldsymbol{\delta}}) - \dim(\boldsymbol{\delta}^*)$ elements of $\hat{\boldsymbol{\delta}}$ if $\dim(\hat{\boldsymbol{\delta}}) > \dim(\boldsymbol{\delta}^*)$.

We evaluate the accuracy of topology inference by a version of graph edit distance [24] that allows merging/splitting nodes. Graph edit distance is a typical performance metric for topology inference algorithms [25]. In our case, a common error is duplicating the same node (i.e., queue) or incorrectly merging different nodes due to errors in the estimated δ-parameters, which is captured by this graph edit distance. As illustrated in Fig. 5, the inferred topology (Fig. 5 (b)) has an edit distance of 1 to the ground truth (Fig. 5 (a)), as merging q_{11} and q_{12} will make it identical to the ground truth. The multicast tree (Fig. 5 (c)) has an edit distance of 3 to the ground truth, requiring d_4 to be split once and d_5 to be split twice.

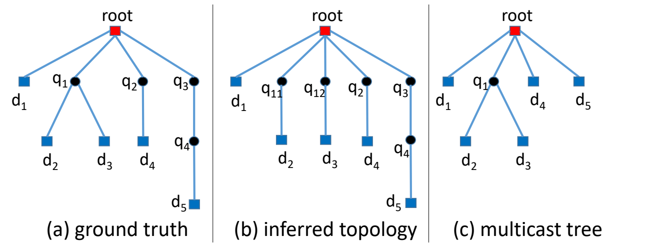


Fig. 5. Example: (b) has edit distance 1 to (a); (c) has edit distance 3 to (a).

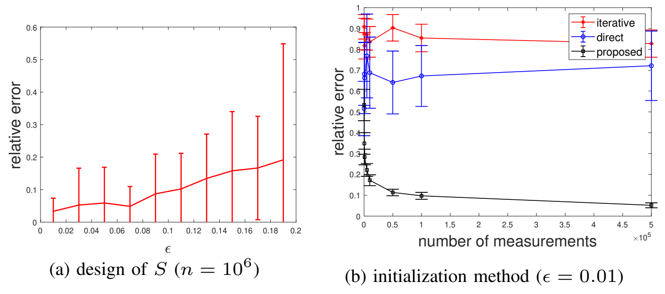


Fig. 6. Evaluation of design choices ($K^* = K = 3$, 100 Monte Carlo runs)

B. Results on Parameter Estimation

We first evaluate the proposed estimator against benchmarks for inferring the δ-parameters of a single path.

Evaluation of design choices: We first evaluate the step size ϵ for set $S = \{0, \epsilon, 2\epsilon, \ldots, 1\}$, at which to fit the empirical Laplace transform. As shown in Fig. 6 (a), the smaller the step size ϵ, the smaller the estimation error, as the estimator will have more data points to fit the Laplace transform. This is, however, at the cost of increased running time, as the complexity of evaluating the objective function (4a) is proportional to $|S|$. Moreover, we evaluate the proposed two-step method of solving (4) as in lines 6–8 of Algorithm 1 against only performing the iterative initialization in lines 6–7 ('iterative') or directly optimizing (4) from an arbitrary initial value ('direct'). The result, in Fig. 6 (b), shows that the proposed method significantly improves the estimation accuracy.

Comparison with benchmarks: We compare the proposed estimator against benchmarks in a variety of settings as shown in Fig. 7–8. MLE is much slower than others (e.g., taking over 15 minutes at $n = 10^5$ compared to 0.02 seconds for others) and hence has fewer Monte Carlo runs. We see that: (i) all the Laplace-based estimators ('proposed', 'Harris 1/2', 'squared') outperform MLE, indicating the advantage of our approach based on the Laplace transform; (ii) directly optimizing the parameter of interest (i.e., $\boldsymbol{\delta}$) to fit the Laplace transform (as in 'proposed' and 'squared') outperforms first estimating the Laplace transform and then inferring the corresponding parameter of interest (as in 'Harris 1/2'); (iii) minimizing the absolute error ('proposed') slightly outperforms minimizing the squared error ('squared') as K^* increases. Furthermore, we see from Fig. 7 that as K^* increases, the errors of all the estimators increase significantly, indicating a limitation on the path length for achieving reasonable accuracy. Meanwhile, comparing Fig. 8 with Fig. 7 (a) shows that the estimators based on Algorithm 1 ('proposed', 'squared') are robust against the presence of virtual queues (i.e., $K^* < K$), validating our idea of using the fitting error to estimate

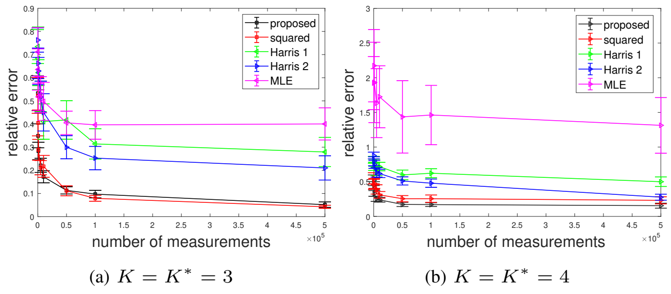


Fig. 7. Comparison under different #queues ($S = \{0, 0.01, \ldots, 1\}$, 20 Monte Carlo runs for MLE, 100 runs for the others).

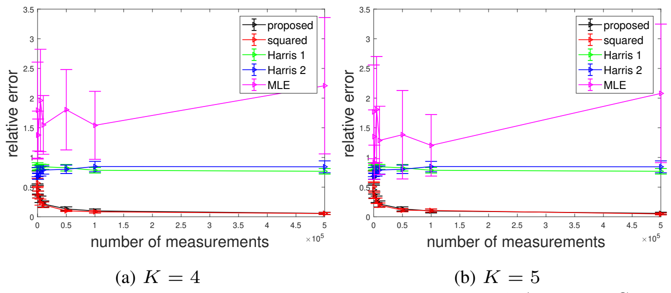


Fig. 8. Comparison under different #virtual queues ($K^* = 3$, $S = \{0, 0.01, \cdots, 1\}$, 20 Monte Carlo runs for MLE, 100 runs for the others).

the actual number of queues (lines 9–10 in Algorithm 1).

C. Results on Topology Inference

Having validated the proposed estimator, we now evaluate the accuracy of using its outputs to infer the topology. All the results are based on 20 Monte Carlo runs.

Impact of threshold Δ and minimum gap Δ^*: Given the estimated δ-parameters, the remaining design parameter is the threshold Δ for detecting the parameters associated with the same queue. According to Theorem IV.1, we want $\Delta < \Delta^*/2$, where Δ^* denotes the minimum gap between the parameters of different queues. Our experiments with various Δ values (plots omitted due to space limitation) also indicated that a small Δ (e.g., $\Delta = 0.01$) works well in general.

Intuitively, our approach will work better if the δ-parameters for different queues are more separated, i.e., when the minimum gap Δ^* is larger. However, as shown in Fig. 9, our algorithm is actually not sensitive to Δ^*. Moreover, for small topologies as evaluated here, the finite-sample accuracy of our algorithm can even beat the asymptotic accuracy of existing multicast-based algorithms (e.g., RNJ [5]). This is because instead of only using low-order statistics (e.g., delay variances and covariances [5]), we examine the entire (empirical) distribution of measurements, which allows us to discover the physical topology that may contain more internal nodes than the (logical) multicast tree.

Impact of tree size: We vary the size of the ground truth topology by separately varying the width of the tree (indicated by the number of destinations N) and the height of the tree (indicated by the maximum path length K), as shown in Fig. 10. As expected, growing either of these parameters will increase the inference error, as the ground truth topology becomes more complex. Compared with solutions designed to infer the multicast tree, we see that our algorithm achieves

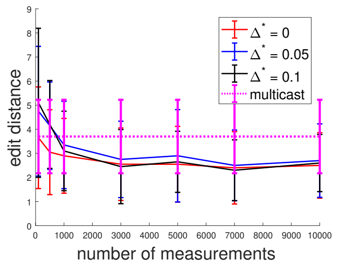
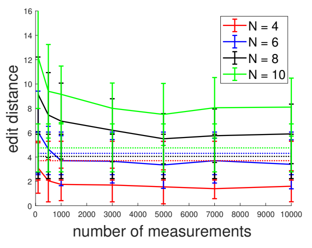
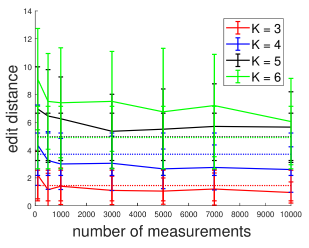


Fig. 9. Impact of minimum gap Δ^* ($K = 4$, $N = 5$, $\Delta = 0.01$, solid: edit distance for inferred topology, dotted: edit distance for multicast tree).



(a) vary N ($K = 4$)



(b) vary K ($N = 5$)

Fig. 10. Impact of tree size ($\Delta^* = 0$, $\Delta = 0.01$, solid: edit distance for inferred topology, dotted: edit distance for multicast tree).

a better accuracy for small topologies ($K \leq 4$, $N \leq 6$) due to its capability of inferring more internal nodes (degree-2 nodes), but becomes less accurate for larger topologies due to the difficulty of estimating the parameters of long tandems of queues and distinguishing the parameters of many queues. We believe this is a fundamental limitation due to the lack of measurements with controllable correlation across paths.

Discussion: We note that the limitation on K is likely to be acceptable in practice as real networks tend to have small diameters, e.g., in AS6461, a single tree of height 4 (i.e., $K = 4$) can cover 83% of nodes. The limitation on N may be addressed by "stiching" trees inferred for small subsets of destinations as in [8], or a hybrid approach that combines our solution with occasional multicast probing. A detailed investigation of these directions is left to future work.

VI. Conclusion

We revisited a classic problem of inferring a tree topology from end-to-end measurements originated by a single source, through a fundamentally different approach that utilizes the detailed queuing dynamics inside the network. Compared to the classical approaches based on multicast or its approximations, our approach has the advantages that it can utilize passive measurements and potentially recover the physical topology with more hidden nodes (degree-2 nodes). Our solution includes a novel estimator that infers the residual capacities of a tandem of queues from end-to-end delays, and efficient algorithms that use the inferred residual capacities to identify the queues shared between paths and infer the queuing network topology. Our solution was guaranteed to be asymptotically consistent, and even outperformed existing solutions based on active probing for small topologies. Meanwhile, we also identified potential directions for further improvements.

References

[1] R. Caceres, N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Loss-based inference of multicast network topology," in *IEEE CDC*, 1999.

[2] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, January 2002.

[3] N. G. Duffield and F. L. Presti, "Network tomography from measured end-to-end delay covariance," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 978–992, December 2004.

[4] N. G. Duffield, J. Horowitz, and F. L. Presti, "Adaptive multicast topology inference," in *IEEE INFOCOM*, 2001.

[5] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang, "Efficient and dynamic routing topology inference from end-to-end measurements," *IEEE/ACM Transactions on Networking*, vol. 18, no. 1, pp. 123–135, February 2010.

[6] M. Coates, R. Castro, M. Gadhiok, R. King, Y. Tsang, and R. Nowak, "Maximum likelihood network topology identification from edge-based unicast measurements," in *ACM SIGMETRICS*, June 2002.

[7] P. Sattari, C. Fragouli, and A. Markopoulou, "Active topology inference using network coding," *Physical Communication*, vol. 6, pp. 142–163, March 2013.

[8] P. Sattari, M. Kurant, A. Anandkumar, A. Markopoulou, and M. G. Rabbat, "Active learning of multiple source multiple destination topologies," *IEEE Transactions on Signal Processing*, vol. 62, no. 8, pp. 1926–1937, April 2014.

[9] H. Yao, S. Jaggi, and M. Chen, "Passive network tomography for erroneous networks: A network coding approach," *IEEE Transactions on Information Theory*, vol. 58, no. 9, pp. 5922–5940, September 2012.

[10] M. Hirabaru, "Impact of bottleneck queue size on tcp protocols and its measurement," *IEICE transactions on information and systems*, vol. 89, no. 1, pp. 132–138, 2006.

[11] D. Katabi and C. Blake, "Inferring congestion sharing and path characteristics from packet interarrival times," *Mass. Inst. Technol., Cambridge, MA, MIT-LCS-TR-828*, 2001.

[12] W. Wei, B. Wang, D. Towsley, and J. Kurose, "Model-based identification of dominant congested links," in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, 2003, p. 115–128.

[13] F. Baccelli, B. Kauffmann, and D. Veitch, "Inverse problems in queueing theory and Internet probing," *Queueing Systems*, vol. 63, p. 59–107, 2009.

[14] F. Pin, D. Veitch, and B. Kauffmann, "Statistical estimation of delays in a multicast tree using accelerated EM," *Queueing Systems*, vol. 66, p. 369–412, 2010.

[15] A. Asanjarani, Y. Nazarathy, and P. K. Pollett, "Parameter and state estimation in queues and related stochastic models: A bibliography," 2017.

[16] D. Gross, J. F. Shortle, and C. M. Harris, "Fundamentals of queuing theory. fourth."

[17] L. Esparza, "Maximum likelihood estimation of phase-type distributions," Ph.D. dissertation, 2011.

[18] H. L. Van Trees, *Detection, Estimation, and Modulation Theory*. John Wiley & Sons, 2004.

[19] J. Abate and W. Whitt, "Numerical inversion of Laplace transforms of probability distributions," *ORSA Journal on Computing*, vol. 7, no. 1, pp. 38–43, 1995.

[20] C. M. Harris and W. G. Marchal, "Distribution estimation using Laplace transforms," *INFORMS Journal on Computing*, vol. 10, no. 4, pp. 448–458, 1998.

[21] A. V. Den Boer and M. Mandjes, "Convergence rates of Laplace-transform based estimators," *Bernoulli*, vol. 23, no. 4A, pp. 2533–2557, 2017.

[22] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *ACM SIGCOMM*, August 2002.

[23] J. Lagarias, J. Reeds, M. Wright, and P. Wright, "Convergence properties of the Nelder-Mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998.

[24] P. Bille, "A survey on tree edit distance and related problems," *Theoretical Computer Science*, vol. 337, no. 1-3, pp. 22–34, 2005.

[25] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, "Looking Glass of NFV: Inferring the structure and state of NFV network from external observations," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1477–1490, 2020.