PAINT: Path Aware Iterative Network Tomography for Link Metric Inference

PAINT: Path Aware Iterative Network Tomography for Link Metric Inference

Leyang Xue†, Mahesh K. Marina†, Geng Li⋆, Kai Zheng⋆
The University of Edinburgh†, 2012 Labs, Huawei Technologies⋆
leyang.xue@ed.ac.uk, mahesh@ed.ac.uk, ligeng66@aliyun.com, kai.zheng@huawei.com

***Abstract*—Understanding link-level performance is key to assuring the quality of cloud-based and OTT services, optimal path selection, robust network operations and beyond. However, direct measurement of each link not only incurs high overhead at the Internet-scale but also is infeasible due to lack of access to network measurement information beyond AS boundaries and functional limitations at relay nodes. Although network tomography is well suited, existing approaches are insufficient due to their unrealistic assumptions with respect to stability, controllability, and visibility. Motivated by this, we propose PAINT, an online iterative algorithm that estimates and refines link-level performance metrics based on path-level measurement. In PAINT, the link metrics are iteratively estimated by minimizing their least square error (LSE) and calibrated based on the comparison of weight between the estimated shortest paths (SPs) and best-known paths from end-to-end path measurements. The key insight is that when there is inconsistency between these paths, then weights of links on the estimated SP are likely misestimated, triggering a further round of estimation to refine the estimated link metrics. Evaluation of PAINT, focusing on link delay estimation, using four different real network topologies and two real-world measurement datasets (including one we collected) shows that relative to existing approaches, it yields up to 3x gain in absolute link delay estimation accuracy and improves decisions dependent on link delay estimation by up to 5x in relative error.**

I. Introduction

Recent years have witnessed cloud service providers expanding their footprint to the edge with their private wide area networks that peer with the public Internet and connecting to a large number of Autonomous Systems (ASs) (e.g., Internet Service Providers (ISPs)) [1], [2]. Similarly, Over the Top (OTT) vendors are delivering their services via overlay networks of their own (e.g., Zoom, Agora SD-RTN™ [3]) and interconnecting with other ASs. These infrastructures serve millions of users every day making user experience and application performance critical concerns for the service providers. To this end, having insight into link-level performance is key. It enables service quality assurance and maintaining robust network operations via traffic engineering (path selection, multi-path load balancing), network troubleshooting (fault identification and localization) and so on (e.g., [4]).

However, directly measuring each link through active probing is not a viable option. At Internet scale, it would cause enormous overhead. Moreover, many administrative entities involved prevent exchanging and accessing operational information across AS boundaries. Some relay nodes also do not allow link measurement. A case in point is the selective forwarding unit (SFU), a key building block to realize real-time communication (RTC) for multiplexing video streams across users at a global scale, that only implements the forwarding functionality [5], [6]. Thus, reliable link metric estimation methods are needed that scale well and cope with limited scope of active in-network measurements.

Network tomography [7]–[9] is a paradigm well suited for this purpose as it allows inferring link-level performance with a limited number of end-to-end path measurements. Specifically, link metrics can be estimated with network tomography through a system of linear equations (SLE) between path measurements and link-level metrics, provided there are sufficient number of independent path measurements (elaborated further in §II-A). However, existing approaches from the network tomography literature (e.g., [10]–[14]) are insufficient as they make certain assumptions contrary to the characteristics and constraints of our target setting: 1) *unstable*: link metrics vary with changes in the choice of peering and intra-domain traffic engineering in third-party AS; 2) *uncontrollable*: monitors (the entities that send and receive measurements) cannot be freely placed on any node and the measurement path cannot be predetermined; 3) *invisible*: intra-domain routes may not be accessible, making the cause of link metric variations latent to the measurement/estimation process.

Motivated by the above, we propose a novel online iterative network tomography algorithm termed "PAINT" that takes a top-down approach, leveraging an application-oriented view for estimating the latent link metrics. Our approach is guided by the following key observation: estimating link metrics is an essential prerequisite for optimal path selection not only with conventional shortest path (SP) algorithms [15] for choosing paths for each node pair but also for more sophisticated traffic engineering with load balancing. We leverage this observation in the design of PAINT: given that computation of optimal paths is a key application scenario for link metric estimation, it can be brought in as a constraint for optimizing the error in the estimation of link metrics. In that sense, the degree to which end-to-end (E2E) SPs are correctly estimated can aid in the link metric estimation process by augmenting the base SLE with additional SP guided equations.

Concretely, PAINT consists of three modules in a closed loop iterative procedure (Fig. 3): estimation, validation and calibration. The estimation module computes link metric estimates through optimization with twin objectives: i) estimate

978-1-6654-8234-9/22/$31.00 ©2022 IEEE

absolute values of link metrics by minimizing least square error (LSE); ii) minimize the difference between estimated shortest paths and known shortest paths from measurements. The validation module checks for inconsistencies by comparing the estimated shortest paths to current best-known path from input path measurements. The difference between estimated SP and observed SP weights are then fed to the calibration module, which identifies links whose metric estimates need to be adjusted and labels them underestimated or overestimated. These labels serve as a guide for link metric estimation in the next round. The above three-step process is repeated until there is no need to adjust any link metric estimates.

We evaluate PAINT against a range of existing approaches (discussed in §II-B) in terms of their relative accuracy and runtime complexity, considering delay as the link metric. We also assess PAINT with respect to these baselines from the perspective of two downstream use cases: (i) pairwise SP estimation leveraged within PAINT; and (ii) estimation of minimal spanning tree (MST) that is commonly employed for cost efficiency. A unique aspect of our evaluation (§IV) is the use of four different real-world network topologies and two link delay measurement datasets. One of these measurement datasets is publicly available while we have collected the other through a cloud service provider from its different vantage points (across multiple continents).

In summary, we make the following key contributions:

- By associating the network tomography problem of link delay estimation from path measurements with the application needs of finding pairwise best quality paths, we propose PAINT (§III), a novel online iterative solution that simultaneously minimizes estimation errors for link metrics as well as pairwise shortest paths. Crucially, inferred link metrics with PAINT are more generally applicable beyond just shortest path computation.
- PAINT constitutes a unique and significant contribution to the network tomography literature (§II-B). Compared to statistical inference approaches like [10], [11], PAINT does not make any assumptions on the stability of link metrics nor does it need to know link metric distributions. In contrast with algebraic and graphical approaches (e.g., [12], [14], [16]), we achieve robust and accurate link metric estimations while avoiding the requirements for controllable measurements and intra-domain route visibility.
- Unlike any prior related work, we use two large real-world measurement datasets and four different real network topologies to conduct evaluations of PAINT (§IV). Our results show that compared to the baselines PAINT achieves 1.5–3x gain in mean absolute error (MAE) and yields 2–5x gain in quality for the typical downstream applications considered.

II. Background

A. Problem Statement

We model the network topology as a directed graph $\mathsf{G} = (\mathsf{V}, \mathsf{E}, \mathbf{w})$, where V, E and $\mathbf{w}$, respectively, refer to the sets of nodes, links and link weights in the graph. Depending on the target scenario, links E can be physical or overlay links in the network, and can belong to the public Internet or private wide area network (WAN) of cloud/OTT service providers. Let $m := |\mathsf{V}|$ be the number of nodes and $n := |\mathsf{E}|$ be the number of links in G. Table I summarizes the notation used in this paper (following the convention in [17], [18]).

TABLE I: Summary of notation. Bold symbol represents matrix or vector.

Symbol	Meaning
G, V, E, $\mathbf{w}$	graph G with V nodes, E links and $\mathbf{w}$ link weights
m, n	number of nodes, links in G
$\lvert\mathbf{A}\rvert$	number of elements in set $\mathbf{A}$
$\mathbf{R}, \mathbf{p}, \mathbf{w}$	binary routing matrix $\mathbf{R}$ (observed), path measurements $\mathbf{p}$ (observed), link weights $\mathbf{w}$ (unknown)
γ	The number of path measurements (rows in $\mathbf{R}$)
$A, \hat{A}$	true, estimated value of variable A
$\mathcal{P}(v_i, v_j)$	path in G between $(v_i, v_j) \in \mathsf{V}^2$
$\mathcal{P}^{\mathbf{w}}(v_i, v_j)$	$\mathbf{w}$ weighted shortest path between $(v_i, v_j) \in \mathsf{V}^2$

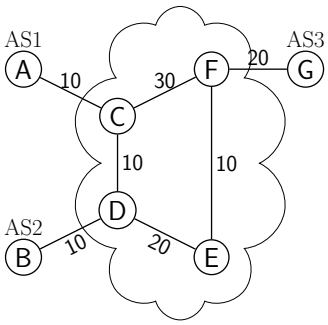
The problem we consider is essentially the estimation (inference) of link weights given a set of end-to-end path measurements. Here the weight of a link corresponds to the metric of interest (e.g., delay, loss rate). Each end-to-end measurement over a path $\mathcal{P}$ in G (e.g., round-trip delay) provides a measurement of the path weight $\mathsf{W}(\mathcal{P}) = \sum w_k, \forall e_k \in \mathcal{P}$ where w_k's are unknown and to be estimated. Multiple measurements over the same path are aggregated through averaging. We assume that paths considered for measurements (both end-nodes as well as paths between those end-nodes) are randomly chosen. We further make the following two common and reasonable assumptions as in the network tomography literature [16], [19]: 1) network topology is known and does not change during the measurement and link metric inference process; 2) measured paths are all simple paths (without cycles) in G; Crucially, note that we do not make any assumptions on the knowledge of distributions from which link weights come from.

We can formally state the problem through a system of linear equations (SLE) relating path measurements with the link weights to be inferred, as in Eq. (1):

$$\mathbf{p} = \mathbf{R}\hat{\mathbf{w}} \tag{1}$$

Here $\mathbf{p}$ denotes the column vector of all observed path measurements, whereas $\mathbf{R} = (R_{ij})$ is a $\gamma \times n$ binary routing matrix, indicating whether link e_j is in the path $\mathcal{P}_i$. Furthermore, row $R_{i\cdot}$ is a sparse representation of $\mathcal{P}_i$, as illustrated in Fig. 1. In this example, a set of path measurements over the network (on top left) are shown (on top right) along with the corresponding binary routing matrix $\mathbf{R}$ (bottom). For instance, the path $\mathcal{P}_1(B, G)$ that goes through links (BD, DE, EF, FG) induces a binary row vector in $\mathbf{R}$. Other chosen paths are similarly represented in $\mathbf{R}$.

With this formulation, resolving our problem of inferring link metrics from path measurements translates to computing $\hat{\mathbf{w}} = \mathbf{R}^{-1}\mathbf{p}$. This requires the matrix $\mathbf{R}$ to be *full rank*, i.e., $\text{rank}(\mathbf{R}) = n$, where each row in $\mathbf{R}$ represents a simple path measurement that is linearly independent from other paths in G. Having the rank of $\mathbf{R}$ less (greater) than n results in an under-determined (over-determined) SLE with infinitely



$(B,G) \mapsto$
60ms (BD, DE, EF, FG)
70ms (BD, DC, CF, FG)
$(A,G) \mapsto$
60ms (AC, CF, FG)
$(A,B) \mapsto$
30ms (AC, CD, DB)

$$\mathbf{R} = \begin{array}{c} \\ \mathcal{P}_1(B,G) \\ \mathcal{P}_2(B,G) \\ \mathcal{P}_3(A,G) \\ \mathcal{P}_4(A,B) \end{array} \begin{array}{c} \begin{array}{ccccccc} AC & BD & CD & CF & DE & EF & FG \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Fig. 1: Example network with relay nodes (C,D,E,F) through which end-nodes (A,B,G) are connected. Each end-node can be an AS or grouped IP prefix. Link labels represent their *true* delay (in ms). Selected path measurements (between end-nodes) over this network are shown on the top right.

many (no) solutions [11], [12], [20]. It is, however, non-trivial to ensure a full-rank $\mathbf{R}$, since n linearly independent measurement paths may not exist or can be difficult to discover with randomly chosen paths for measurements.

Going back to the example in Fig. 1, there are 7 variables (link weights) to solve but only 4 equations. In this case, $\text{rank}(\mathbf{R}) = 4$, which means SLE is rank deficient and so has infinitely many solutions (i.e., no unique solution.). Even when every path between end-nodes is measured in this example, there are still 6 equations for 7 variables, leading to an under-determined linear system. This illustrates the difficulty of finding a straightforward algebraic solution for $\hat{\mathbf{w}}$.

B. Related Work

Network tomography generally aims at recovering performance metric of links from end-to-end path measurements for general topologies [7]. By establishing a functional relation between link metrics and path measurements, links metrics are approximated by evaluating the inverse of the function, as outlined in §II-A. As illustrated in Fig. 2, existing analog tomography (real numbered estimation of link metrics) approaches fall into three major categories: statistical, algebraic and graphical. Statistical approaches typically assume prior knowledge of link metric distribution to approximate the solution. Algebraic approaches use matrix properties and apply linear algebra methods to obtain an exact solution. Graphical approaches adopt spanning tree and connectivity properties to construct independent measurement paths, to enable algebraic solutions.

Statistical approaches have two major threads: 1) estimating values of link metrics (e.g., [10]); 2) estimating distribution parameters of link metrics (e.g., [11]). Bu et al. [10] have first brought discretized modeling to Expectation–Maximization (EM) for model simplification and run-time optimization. Liang and Yu [21] apply the idea of divide and conquer on

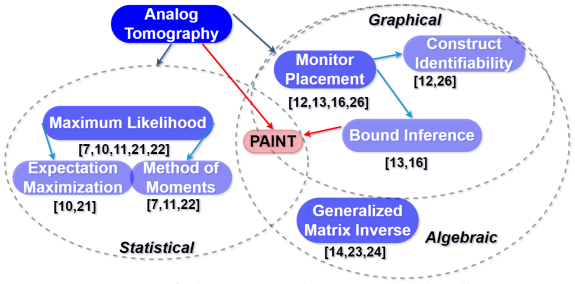


Fig. 2: Taxonomy of the network tomography literature. Color transparency indicates the onward development of algorithms (from dark to light). Dashed line circles indicate different categories of approaches. References for each (sub-)category shown as labels.

Maximum Likelihood Estimation (MLE) with latent variable by solving smaller sub-problems. Later, Chen et al. [11] target heterogeneous metrics with an adaptive model and distribution approximation in Fourier domain using General Method of Moments (GMM). Deng et al. [22] take spatial dependency of multicast tree into consideration for estimating distribution parameters.

Within *algebraic* approaches, broadly there exist two kinds of methods: 1) solve SLE using numerical matrix approximation, i.e., Moore–Penrose inverse (pseudoinverse) (e.g., [14]); 2) solve SLE directly when it has full rank. The problem of finding pseudoinverse is well studied and is often solved by matrix factorization. Chen et al. [23] propose a QR decomposition based solution for general topologies. Considering the sharing of links between paths, Chua et al. [24] apply singular value decomposition (SVD) to the same problem and use it in a monitoring system. Song et al. [14] apply both the above algorithms to measurement system for path selection.

On the other hand, the sub-category of *graphical* approaches within algebraic approaches aim at obtaining a full rank SLE. As that is an NP-hard problem under uncontrollable routes [25], graphical approaches introduce controllability to simplify the problem [26]. Specifically, the goal here is to maximize the rank of SLE under two distinct scenarios: 1) given full controllability, producing full rank system using minimal number of monitors and measurement paths (e.g., [12]); 2) given only path controllability, use minimal number of paths to achieve tighter estimation bounds for unidentifiable links (e.g., [16]). Bejerano et al. prove that identifying all links with uncontrollable routes is an NP-hard problem [25] so they simplify the problem by using controllable route as hybrid approach leveraging both graph information (e.g., spanning tree, connectivity) and SLE properties (value positiveness, reduced form of matrix). Ma et al. [12] resolve that problem efficiently using fully controllable monitor placement and routes. Feng et al. [16] estimate the value bound for unidentifiable links given a fixed set of monitors and tighten bound with random new monitor placements. Li et al. [13] provide tighter value bounds for unidentifiable links with more carefully designed monitor placement.

While the above discussed approaches each represent an advance in the network tomography literature, they have

three main shortcomings when applied for monitoring and optimizing quality of cloud and OTT services today:

- **Stability:** To estimate distribution parameters, statistical approaches (e.g., [22]) assume that link metrics exhibit stable behavior for the duration of path measurement collection. Graphical approaches, on the other hand, rely on the assumption of symmetric link metrics. Such conditions typically do not hold in practice, since paths may be changed due to traffic engineering and for load balancing, causing fluctuations and asymmetries in link metrics (e.g., delay).
- **Controllability:** For constructing linearly independent measurement paths, graphical approaches usually require a single entity that can control all measurements, e.g., using source routes [12], [16]. Such requirement may not be possible to meet due to operational constraints and security concerns. In addition, monitor placement in the data plane might be unavailable on UDP oriented forwarding nodes, a common scenario for real-time communication.
- **Visibility:** Some statistical approaches require prior knowledge of metric distribution for each link in the network (e.g., [21]). On the other hand, the measurement methodology for algebraic approaches [14] relies on traceroute but this tool is often blocked by network providers to keep their routing strategy confidential.

In contrast, PAINT (as shown in Fig. 2) is a hybrid approach retaining only the best aspects of both statistical and algebraic (graphical) approaches. Compared to statistical approaches, PAINT does not make any assumption on the prior knowledge of link metric distribution nor on the link metric fluctuation. With respect to algebraic and graphical approaches, PAINT relaxes the requirements for symmetric link metrics, controllability and visibility. PAINT can work with arbitrary path measurements, obviating the need for full controllability of paths and monitors. Moreover, shortest path (SP) based regularization in PAINT has better conditioning as SP has greater tolerance to link delay fluctuations. Moreover, prior works from the path-to-link network tomography literature discussed above, although they obtain a unique solution under their respective frameworks, tend to suffer from distribution drift from measurements and also not leverage / correlate with performance for downstream tasks.

While we leverage optimal path selection as a key application of path-to-link network tomography in designing PAINT, the path-to-link network tomography has several other applications. Generally speaking, it can help overcome the restrictions for active measurements to indirectly expand the coverage and granularity of measurement to all links [25]. Path-to-link tomography can also enable efficient estimation of link metrics even when there are no such restrictions [26], [27]. BeCAUSe [28] is an interesting recent work that uses path-to-link tomography to reveal the deployment of route flap damping in BGP routers. Network troubleshooting (identifying and locating faults) is yet another application. For example, BlameIt [29] identifies network segment with faults based on passive end-to-end measurements.

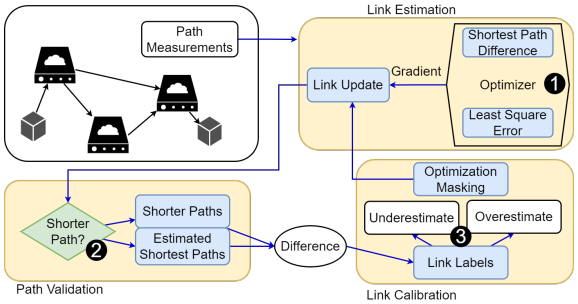


Fig. 3: Schematic representation of PAINT with yellow colored boxes representing its main components.

III. PAINT Algorithm

In this section, we present our proposed path aware iterative network tomography (PAINT) algorithm that estimates shortest paths in tandem by way of optimizing the error in estimating link metrics. For concreteness, we focus on delay as the link metric to be inferred, considering that user experience, especially for interactive and immersive applications, depends on low end-to-end (E2E) latency [29]–[31]. However, the approach we take in PAINT is equally applicable to other link metrics (e.g., loss rate), as we discuss later in this section.

A. Overview

In a nutshell, PAINT estimates link metrics from end-to-end path measurements with the help of graphical regularization, driven by the intuition that solely minimizing the error in fitting SLE is in general insufficient (as elaborated in §II-A) and also may not be beneficial for downstream applications. Specifically, we propose a shortest path (SP) based regularization that considers the relative magnitude among link metrics estimations, besides the typical optimization focusing on estimating the absolute value of link weights. It is indeed quite natural to couple the link metric estimation algorithm with the key application scenario of finding optimal paths, so that decision quality concerning SP benefits from the higher accuracy in network tomography based link weight estimation and vice versa.

While our intuition may seem simple, translating this into a solution is far from trivial. To see this, note that accurate estimations of link metrics are a prerequisite for deciding best (shortest) paths (SPs) for every end-to-end node pair. In our context, SP refers to minimum weight (e.g., delay) path, where path weight is the sum total of weights of links making up the path. The fact that link metrics are latent with respect to available end-to-end path measurements makes it impossible to calculate actual SPs.

To realize the above idea, as illustrated in Fig. 3, PAINT algorithm initially takes path measurements as input to a three-step iterative procedure, as outlined below. (❶) *Estimation*: this step performs optimization with twin objectives: i) estimate absolute values of link metrics by minimizing least square error; ii) minimize the difference between estimated shortest paths and known shortest paths from measurements.

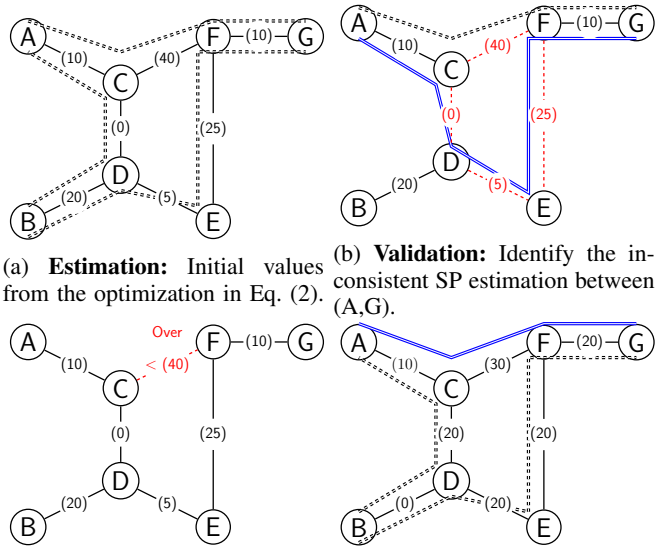
(❷) *Validation*: this step validates if the estimated SPs based on the link weight estimations from the estimation module are optimal by comparing them against SP weights from path measurements. Any inconsistencies found provide evidence for the following calibration step to trigger another round of link estimation. (❸) *Calibration*: this step produces labels for those links whose weights are underestimated or overestimated. Only the labeled links are updated in the next round of link estimation.

Fig. 4 illustrates the working of the PAINT algorithm. Note that in the first link estimation step of PAINT, link weights are estimated by minimizing least square error (LSE) and shortest path error (SPE) (as in Eq. (2)). To bootstrap the operation of the algorithm, we obtain initial link weight estimates (Fig. 4a) using SVD [32] rather than naively assigning them to random weights. Initial SP estimation is based on these initial link weights. We choose SVD for initial link weight computation as it not only has a fast runtime but also is robust in terms of minimizing the least square error under insufficient rank [24], [32]. Aside from this initialization aspect, note that PAINT works with end-to-end path measurements between any set of node pairs, even if they do not result in a full rank SLE. This essentially makes PAINT free from the "controllability" requirement that some prior approaches have.

The following validation step (Fig. 4b) finds the estimated SP between A and G to be inconsistent with the observed SP between those nodes based on path measurements (from Fig. 1). The other estimated SPs between A and B, and between B and G are however validated to be correct. Note here that for validation, as the actual SPs are unknown, the algorithm uses the best known paths from measurements as reference and without knowledge of link metric distributions, thereby obviating the need for "visibility". The following calibration step labels link CF as overestimated (Fig. 4c). This step not only helps address the "stability" issue by adapting to link metric fluctuations but also to efficiently handle link metric mis-estimation by focusing on links that negatively affect the quality of selected paths. In the subsequent round of link estimation (Fig. 4d), the weight of link CF is adjusted down, which results in all SPs to be correctly estimated as well as lowering the absolute error of individual link weight estimates.

B. Detailed Design

1) Estimation: Given the measurement $\{\mathbf{R}, \mathbf{p}\}$, the goal of estimation module is to provide accurate link estimation by minimize both LSE and SPE as in Eq. (2). The first part of the equation is LSE: the Euclidean distance between path measurements and the corresponding estimated path metrics. The second part represents SPE: the sum of differences in cost between estimated SP ($\mathcal{P}^W_{\text{est}}$) and corresponding reference SP ($\mathcal{P}^W_{\text{ref}}$), denoted as $\Delta(v_i, v_j)$ in Eq. (2), for each distinct node pair in $\mathbf{p}$. The reference SP for a node pair is essentially the *observed* shortest path for that node pair. To prevent the optimization in Eq. (2) from falling into a false minima (e.g., link metrics underestimated to minimize the SPE), we scale



(a) **Estimation:** Initial values from the optimization in Eq. (2).

(b) **Validation:** Identify the inconsistent SP estimation between (A,G).

(c) **Calibration:** Determine the link metrics to be calibrated.

(d) **Estimation:** Repeat estimation of labeled link weights.

Fig. 4: Example illustrating PAINT algorithm operation. Double dashed (black) lines indicate actual end-to-end SP as in Fig. 1. Labels on the links indicate estimated link metric (delay) values. Double solid (blue) lines indicate estimated SP. Dashed single (red) lines indicate links on which estimated metrics should be adjusted.

SPE by λ so that the second part in Eq. (2) also influences the optimization. Specifically, we set λ to $\|\hat{\mathbf{w}}\|/(\gamma\bar{\mathbf{p}})$, given that there are γ paths in $\mathbf{p}$, each with a weight $\bar{\mathbf{p}}$ on average.

$$\min_{\hat{\mathbf{w}}} \left\{ \|\mathbf{p} - \mathbf{R}\hat{\mathbf{w}}\| + \lambda \sum_{(v_i,v_j)\in\mathbf{p}} \Delta(v_i, v_j) \right\}, \qquad (2)$$

$$\Delta(v_i, v_j) = \mathsf{W}\left(\mathcal{P}^{\hat{\mathbf{w}}}_{\text{est}}(v_i, v_j)\right) - \mathsf{W}\left(\mathcal{P}^{\mathbf{w}}_{\text{ref}}(v_i, v_j)\right)$$

Overall, the above optimization works like gradient descent, while selectively updating link metric estimations across iterations. As the ground truth SPs are latent, we only have reference SPs ($\mathcal{P}^W_{\text{ref}}$) from path measurements to compare with. Our key idea here is to have reference SPs serve as regulators to help us get closer to the ground truth in terms of SP weights. In other words, while updating $\hat{\mathbf{w}}$ during the optimization, new paths with a smaller (or equal) weight than the corresponding measured paths are discovered. What is described above refers to an iteration of the link estimation step. In the following iteration, only a subset of the links that are identified and labeled to be mis-estimated (respectively, by the validation and calibration steps) are selected for metric estimation update during the optimization.

2) Validation: Given the estimation $\hat{\mathbf{w}}$ from estimation module and path measurements $\{\mathbf{R}, \mathbf{p}\}$, we need to determine whether another iteration to update the link metric estimates is needed. That is the task of this validation step. It is made up of two parts: 1) check whether each link's metric estimate has a valid value; 2) check whether each SP weight estimate has a valid value with respect to the corresponding reference SP. The links that individually fail the validation check 1), or

$$\mathbf{M} = \text{RREF}([\mathbf{R}|\mathbf{p}]) = \left[\begin{array}{cccc|ccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 10 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 60 \\ 0 & 0 & 1 & 0 & -1 & -1 & -1 & -40 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 50 \end{array}\right]$$

$$\implies \begin{pmatrix} w_1(AC) \\ w_2(BD) \\ w_3(CD) \\ w_4(CF) \end{pmatrix} = \begin{pmatrix} 10 \\ 60 \\ -40 \\ 50 \end{pmatrix} - \mathbf{M}_f \mathbf{w}_f \geq 0,$$

$$\mathbf{M}_f = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ -1 & -1 & -1 \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{w}_f = \begin{pmatrix} w_5(DE) \\ w_6(EF) \\ w_7(FG) \end{pmatrix}$$

$$\implies \mathbf{M}_f \mathbf{w}_f = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ -1 & -1 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_5(DE) \\ w_6(EF) \\ w_7(FG) \end{pmatrix} \leq \begin{pmatrix} 10 \\ 60 \\ -40 \\ 50 \end{pmatrix}$$

Fig. 5: Example $\{\mathbf{p}, \mathbf{R}\}$ for NVB inference.

are part of mis-estimated SPs that fail the validation check 2) need to be considered for update in the subsequent iteration. Note that the first check ensures that links that are not covered by the path measurements are taken into account and updated to limit their estimation errors.

For the first check, we verify if each link's estimate falls within a "natural bound", considering that all link metrics are positive. To this end, we define natural value bounds (NVBs) for the estimation $\hat{\mathbf{w}}$ in Definition 1. The link weights that violates their bounds will need further updates in later iterations.

Definition 1. *Natural value bound (NVB): In an SLE, the natural value bound of a variable $w_i \in \hat{\mathbf{w}}$ is the interval $\mathcal{B}(w_i) = [l_i, u_i]$. Given any set of variables assigned values within their NVBs, the deduced values of all other variables from SLE will always be positive. When all the variables are in such intervals, SLE can be solved exactly or has the least square solution.*

NVB can be derived from SLE using inequalities and Gaussian elimination. Exceeding the bound on one link may result in negative value estimation on other links due to LSE optimization and vice versa. We illustrate the calculation of NVB through an example in Fig. 5 that corresponds to path measurements in Fig 1. Firstly, the augmented matrix $\{\mathbf{R}, \mathbf{p}\}$ is transformed to reduced row echelon form (RREF) via Gauss-Jordan elimination [33]. The resulting $\mathbf{M}$ has 1 as a leading coefficient in each row and can be seen to be made up of three parts: an identity matrix, the reduced coefficients $\mathbf{M}_f$ and the reduced $\mathbf{p}$. Since all link weights are positive, by rearranging the SLE represented by $\mathbf{M}$, we get the following inequality: $\mathbf{M}_f$ times corresponding link weights $\mathbf{w}_f$ must be less than the reduced $\mathbf{p}$, where $\mathbf{M}_f$ and $\mathbf{w}_f$ are as shown in Fig. 5. From this inequality, we get the bounds for three of the weights as $w_5 \in [0, 40], w_6 \in [0, 40], w_7 \in [0, 40]$, while we get $w_1 \in [0, 10]$ directly from $\mathbf{M}$. By substituting the value range for these 4 weights back to the set of linear equations, we obtain the remaining bounds as $w_2 \in [0, 30], w_3 \in [0, 30], w_4 \in [0, 30]$.

Algorithm 1: PAINT Algorithm

Input: Path measurements $\{\mathbf{p}, \mathbf{R}\}$
Output: Final estimation $\hat{\mathbf{w}}$

repeat
  `// Estimation`
  Optimize Eq. (2) focusing on labeled links to get $\hat{\mathbf{w}}$, then calculate all SPs for observed node pairs.;
  `// Validation`
  Create empty link violation set $\mathcal{E}_{nvb}, \mathcal{E}_{sp}$;
  Create empty link non-violation set $\mathcal{E}_n = \{\}$;
  $\forall \hat{w}_i \in \hat{\mathbf{w}}$ if $\hat{w}_i$ is outside its NVB, add e_i to $\mathcal{E}_{nvb}$;
  $\forall (v_i, v_j) \in \mathbf{p}$ if $\mathcal{P}^{\hat{\mathbf{w}}}_{\text{est}}(v_i, v_j)$ violates Eq. (3), add all $e_i \in \mathcal{P}^{\hat{\mathbf{w}}}_{\text{est}}(v_i, v_j)$ to $\mathcal{E}_{sp}$; else add them to $\mathcal{E}_n$;
  `// Calibration`
  $\mathcal{E}_v = \mathcal{E}_{nvb} + (\mathcal{E}_{sp} - \mathcal{E}_n)$;
  If $\mathcal{E}_v$ is a null set, repeat step (6) and set $\mathcal{E}_v = \mathcal{E}_{sp}$;
  $\forall e_i \in \mathcal{E}_v$, label e_i for update during optimization in next iteration. Estimates for all unlabeled links remain unchanged in the next iteration.
until $\mathcal{E}_v$ *is empty*;

After validating against the bounds for each link as above, we validate the estimated SP between each observed node pair by comparing its weight against the corresponding measured path weight. This allows us to place an additional constraint to detect mis-estimations. While the reference SPs from measurements do not reflect the ground truth SPs, they provide a useful bound for assessing the quality of estimated SPs. As such, we perform the check in Eq. (3) and identify the links of all estimated SPs violating this constraint to be mis-estimated.

$$\forall (v_i, v_j) \in \mathbf{p}, \mathsf{W}(\mathcal{P}^{\hat{\mathbf{w}}}_{\text{est}}(v_i, v_j)) \leq \mathsf{W}(\mathcal{P}^{\mathbf{w}}_{\text{ref}}(v_i, v_j)) \quad (3)$$

3) Calibration: The goal of this calibration module is to label links whose estimates need further updates. Here we do not specifically distinguish between underestimation and overestimation, but let the optimization in Eq. (2) to handle it in the next iteration. Formally, the calibration module labels the links that need updates according to the following sequence of rules; the later rules override the former ones. First, all the links failing the NVB validation check are labeled. Second, links of all estimated SPs violating Eq. (3) are labeled. Third, links of estimated SPs satisfying Eq. (3) are unlabeled. Finally, if the set of labeled links is empty then we reapply the second rule. These set of rules not only reduce the link metric estimation errors with each iteration but also allow quick convergence by labeling the fewest number of links for further updates in the subsequent iteration.

4) Complete Workflow: PAINT combines the above described three modules together, as listed in the pseudocode form in Algorithm 1. During estimation (2), we run the optimization in Eq. (2) focusing on labeled links to obtain

updated link metric estimations and compute SPs accordingly. Note that all links are labeled to start with in the first iteration of the algorithm. During validation ((3)–(6)), we verify the bounds for links and SPs, and identify links whose estimates potentially need updates. Lastly, during calibration ((7)–(9)), we apply four rules to narrow down the set of links to be updated and label them for consideration in the next iteration. This whole process is repeated until there are no labeled links from the calibration phase.

C. Algorithm Generalization and Analysis

Although we presented the PAINT algorithm using delay as the link metric for concreteness, it is more generally applicable to both additive and multiplicative metrics. For example, PAINT can be used for link loss rate estimation with the following log transformation using the complementary $\mathsf{W}_{\text{succ}}(\mathcal{P})$:

$$\mathsf{W}_{\text{succ}}(\mathcal{P}) = 1 - \mathsf{W}_{\text{loss}}(\mathcal{P}) = \prod_{e_i \in \mathcal{P}} (1 - \mathsf{W}_{\text{loss}}(e_i))$$

$$\implies \log\left(\mathsf{W}_{\text{succ}}(\mathcal{P})\right) = \sum_{e_i \in \mathcal{P}} \log\left(\mathsf{W}_{\text{succ}}(e_i)\right)$$

Note that with sufficiently large number of diverse path measurements, PAINT is guaranteed to converge to a near-optimal solution in terms of link weight estimates. This is because: (1) such measurements will allow SLE to be full rank and ensure a unique solution for LSE minimization in the first part of our optimization; (2) with sufficiently large number of path measurements, reference SPs ($\mathcal{P}_{\text{ref}}^{W}$) will include the true SPs, which will then be discovered by PAINT through the second part of the optimization in Eq. (2). In other words,

$$\text{As } \gamma \mapsto \infty, \mathsf{W}(\mathcal{P}_{\text{est}}^{\hat{\mathbf{w}}}) = \mathsf{W}(\mathcal{P}_{\text{ref}}^{\mathbf{w}}) = \mathsf{W}(\mathcal{P}_{\text{gt}}^{\mathbf{w}}) \tag{4}$$

In such an ideal case, the most deviation from the optimal solution ($\mu(\mathsf{E})$) can also be well characterized. Specifically, the worst case occurs when link weight estimates differ by a maximum amount that still ensures that SPE in Eq. (2) is zero. This is formally stated in the following remark.

Remark 1. *The worst case error ($\boldsymbol{\epsilon}$) between estimated and true mean of link weights ($\|\hat{\mathbf{w}} - \mu(\mathsf{E})\|$) is the maximal change in $\hat{\mathbf{w}}$ that does not yield any SPE, i.e., $\boldsymbol{\epsilon} \neq 0, \hat{\mathbf{w}} = \mu(\mathsf{E}) + \boldsymbol{\epsilon}$, such that $\forall (v_i, v_j) \in \mathsf{V}^2, \mathsf{W}(\mathcal{P}_{est}^{\hat{\mathbf{w}}}(v_i, v_j)) = \mathsf{W}(\mathcal{P}_{ref}^{\mathbf{w}}(v_i, v_j)) = \mathsf{W}(\mathcal{P}_{gt}^{\mathbf{w}}(v_i, v_j))$.*

For this case, we can also quantify the convergence speed in terms of number of iterations, assuming a constant learning rate α, as given in Corollary 1.1.

Corollary 1.1. *With constant learning rate α, $\forall \boldsymbol{\epsilon} \in \mathbb{R}^n, \hat{w}_i \in \hat{\mathbf{w}}, \mu_{e_i} \in \mu(\mathsf{E})$, at least the following steps will be iterated for convergence.*

$$\frac{1}{\alpha} \max \|\hat{w}_i - (\mu_{e_i} + \epsilon_i)\|$$

Proof. The most optimization steps take place in SPE part of the optimization as it has linear descent in terms of its gradients, which is larger than the quadratic descent in LSE minimization. Since the optimization of all links is computed in parallel, the link with the deepest descent determines the required number of iterations. □

IV. Evaluation

In this section, we evaluate PAINT using real-world topologies and measurement datasets, focusing on delay as the link metric. We aim at assessing *absolute link delay estimation error* as well as *relative decision quality* for two downstream use cases with PAINT in comparison with representative state-of-the-art approaches. We also study the responsiveness of link metric inference with PAINT relative to network dynamics.

A. Datasets

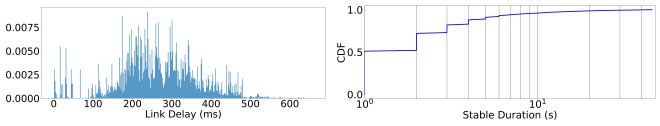
We evaluate PAINT and alternative approaches over four real-world network topologies of different sizes that are commonly considered in prior studies (e.g., [34], [35]): (i) National Science Foundation Network (NSFNET, 14 nodes, 42 links); (ii) German Backbone Network (GBN, 17 nodes, 52 links); (iii) European data network for the research and education community (GEANT2, 24 nodes, 74 links); (iv) German 50 nodes backbone (Germany50, 50 nodes, 176 links).

TABLE II: Summary of measurement datasets. m and n are number of nodes and links in the measurement graph.

	m	n	μ (ms)	σ^2 (ms)	samples
NLANR-AMP	142	9128	$[1, 323]$	$[0, 45897]$	$\sim 10^7$
RTC-CLOUD	87	1971	$[3, 365]$	$[0, 5200]$	$\sim 10^6$

The above topology datasets by themselves are unweighted so the next step is to assign weights to links based on real-world link delay measurement data. To this end, we use two measurement datasets: (1) the publicly available dataset from the National Laboratory for Applied Network Research Active Measurement Project (NLANR-AMP) [36]; (2) a measurement dataset we collected over one of the leading RTC cloud platform (RTC-CLOUD). These two measurement datasets are summarized in Table II, where μ and σ^2, respectively, represent the range of average delays and delay variances across all links. The range of these values reflect the high degree of diversity in terms of both link delays and link delay fluctuations.

We randomly sample links from either the RTC-CLOUD or the NLANR-AMP dataset and map those measurements to links in the four topologies. We use hundreds of trials of such sampling and report results that are averaged across those trials. While both datasets are obtained using active probing for link measurement, we use these measurements in our evaluations without regard to how they are collected (active/passive). As per the measurement paths, we pick them through random walks over the topology in question. Note that estimation errors reported consider links that are covered by at least one measurement path (random walk). The above approach to combining delay measurement datasets and real-world topologies allow us to evaluate PAINT and baselines across a wide range of scenarios.



(a) Delay (b) Stability

Fig. 6: Link delay distribution and stability in RTC-CLOUD.

As the NLANR-AMP dataset is well-known, here we elaborate further on the RTC-CLOUD dataset we collected. It consists of ping measurements (every second) on overlay links between nodes in the dataset. The RTC-CLOUD consists of two types of measurements: edge node to relay node measurements and measurements between relay nodes. The dataset is constructed with 27 relay nodes on 16 sites and 60 edge nodes across several major continents (Europe, Africa, Asia, South America). The relay nodes are VMs in data centers and are interconnected by tunneling through private WAN and public Internet links. Edge nodes connect to relay nodes via WiFi or 4G.

Fig. 6a shows the distribution of link delays in the RTC-CLOUD dataset. We observe that links passing through the public Internet exhibit high delays in the range of 100-500ms. Figure 6b shows the timescales of link delay fluctuations in the RTC-CLOUD dataset. In particular, duration over which different link delays are stable is shown as a CDF. We observe that link delay is stable only within a 1 second period in 50% of the cases. This provides a guideline for how quickly the link delay estimation needs to be done for it to be useful for network control decisions (path selection, etc.).

B. Baselines

We evaluate PAINT in comparison with representative state-of-the-art link metric estimation algorithms from the network tomography literature (reviewed earlier in §II-B). We consider 5 different baseline approaches as outlined below. We consider EM [10] and GMM [11] algorithms to represent the statistical approaches. From the algebraic approaches, we consider SVD [24] and controllable path measurements [16]. From the graphical approaches, we consider two cases as outlined below. The approach requiring full controllability on both monitors and paths yields ground truth as the answer, and as such implicitly considered in our evaluation. Naïve only picks random values from value bounds, reflecting a graphical approach where measurement paths are uncontrollable. Fully Controllable Route (FCR) [16] baseline is an intermediate approach between the above two graphical approaches, and serves as the best upper bound for estimation accuracy and decision quality, although impractical/unrealistic.

PAINT and baselines are run until convergence with results shown reflecting this point. Link metric estimation is computed directly for algebraic approaches (SVD) and graphical approaches (FCR, Naïve). For statistical approaches (EM and GMM), convergence is assumed when the difference between successive estimations falls below 10^{-6}. The convergence of PAINT occurs when the calibration step in an iteration does not label any links as mis-estimated. All the algorithms are implemented with Python NumPy and PyTorch library on a commodity server (CPU: AMD EPYC 7453, MEM: 1000 GB, OS: Ubuntu 20.04, GPU: NVIDIA A5000). PAINT is implemented using GPU based optimizer and matrix operators.

C. Link Delay Estimation Accuracy

We use Mean Absolute Error (MAE) as the metric to quantify link delay estimation accuracy with different algorithms. For each link, closer the estimated link delay is to the ground truth, smaller the value will be. The outliers among MAE across all links also provide the information on the best/worst performance of the algorithms. Note that we calculate the MAE for each link in every trial separately and show the aggregate result.

Fig. 7 shows the MAE for link delay estimation across all links over 100 random trials. We observe that PAINT performs close to the best, similar to the impractical FCR and significantly better compared to statistical and algebraic baselines. We also observed that in a simple network topology, PAINT may outperform FCR. Here we restrict the number of monitors for FCR to make it more challenging to construct full-rank SLE. When full-rank SLE is not available, the unidentifiable links fall back to naive approach, which have significant impact on accuracy.

PAINT achieves 2x to 3x accuracy gain when using NLANR-AMP measurement dataset and 1.5x to 2x gain with the RTC-CLOUD dataset. Comparing to SVD that is purely a least square optimization algorithm, we note that the gain comes mainly from the SP based regularization in PAINT. Since the accuracy is evaluated without historical knowledge, the statistical algorithms lack sufficient data to establish the distribution, especially for EM that uses single distribution for each link. Use of a set of distributions as in GMM is seen to improve the estimation accuracy significantly.

The overall MAE with RTC-CLOUD dataset is larger than that with NLANR-AMP. This can be attributed to the larger link delays in RTC-CLOUD dataset as overlay links are mostly transcontinental. If the number of possible paths between a pair of nodes is huge, it is hard to find a shorter path for reference within initial measurements. In addition, regularization using SP does not pay as much attention to the links with large delay as it is unlikely for them to appear in SPs, leading to higher estimation error for such links.

We also observe that the accuracy of PAINT decreases slightly when the network size becomes larger (left to right). This is mainly because we do not scale up the number of path measurements proportionally with the network size. So for a larger network, coverage of measurements reduces, leading to less regularization effect for individual links. As we will see later in this section, using fewer path measurements for larger networks causes little harm for downstream applications.

Besides having a low MAE, it is also desirable to have fewer outliers in the estimation of link weights. Such outliers may overestimate “good” links or underestimate “bad” links, thus affect the decision quality in downstream applications (e.g., shortest path, choosing top k best paths). To examine the nature of outliers with PAINT, Fig. 8 shows the CDF

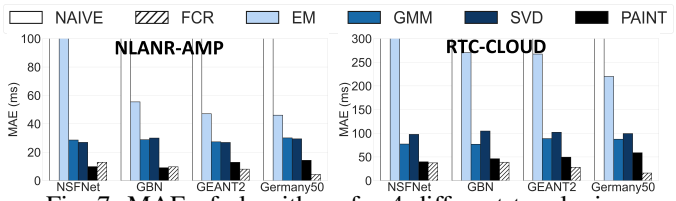


Fig. 7: MAE of algorithms for 4 different topologies.

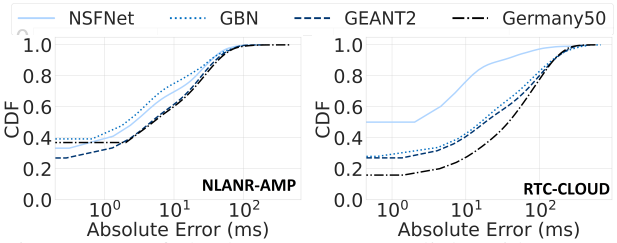


Fig. 8: CDF of absolute errors across links with PAINT.

of absolute errors across links after convergence. We observe that more links are estimated correctly on smaller network sizes due to a higher coverage of path measurements in these topologies. This shows the need to ensure a sufficient coverage of links with path measurements with increasing network sizes, especially to reduce the worst case absolute estimation errors.

D. Runtime Complexity

We evaluate the runtime complexity of the algorithms to demonstrate their ability to adapt to network dynamics. Recall that the link performance fluctuations in the measurement datasets happen at the scale of seconds, whereas system-wide control decisions, e.g., route damping prevention and choice of best path, are usually made every few minutes or even hours [28], [37], [38]. Algorithms with runtime faster than the link delay dynamics will result in more traffic with "good" performance for downstream applications.

Fig. 9 shows the longest runtime for each algorithm across the two measurement datasets for different topologies. We see that Naïve always has the lowest runtime complexity due to its simplistic random assignment of link weight estimates from within value bounds. SVD also has very low runtime due to the underlying parallel matrix computations and efficient operation over multicore processors. Methods from the statistical approaches relatively have higher runtime complexity since they need exhaustive search, especially for GMM that has a higher dimensional search space than EM. We see that PAINT scales well with different network sizes and always runs under a second (within the time period when link delays are seen to be stable). We further note that in these experiments the GPU accelerator is only lightly utilized (at 16%), so there is further potential to scale to much larger networks without increasing the runtime. Although FCR can provide accurate estimation when full control is possible (e.g., in a private WAN) and when the assumption of symmetric links holds, the search for monitor placement makes the algorithm less scalable than others as dynamic programming used in FCR has faster increase in runtime with network size then all other algorithms.

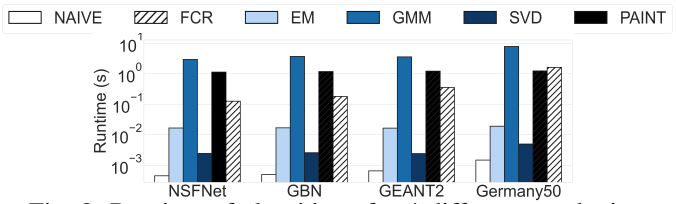


Fig. 9: Runtime of algorithms for 4 different topologies.

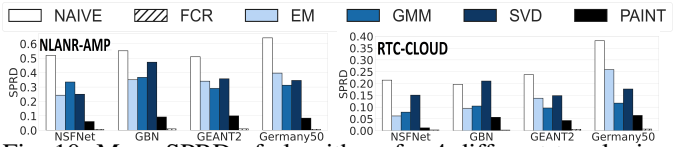


Fig. 10: Mean SPRD of algorithms for 4 different topologies.

E. Shortest Path Quality

As noted at the outset, SP is a key application scenario for link metric estimation, where the aim is to determine the best paths in the network to direct traffic flows. It is typically seen as a sub-problem in traffic engineering, load balancing and ensuring service quality [15], [39]. For this problem, closeness to the optimal path is typically of interest when algorithm produces a group of possible paths satisfying service level requirements [38], as opposed to exact match with the shortest path. Given the above, we use SP Relative Difference (SPRD) as a metric to evaluate the quality of estimated shortest paths, defined as follows. For any ground truth SP $\mathcal{P}^W_{\text{gt}}$ and estimated SP $\mathcal{P}^W_{\text{est}}$, SPRD is equal to $|\mathsf{W}(\mathcal{P}^W_{\text{gt}}) - \hat{\mathsf{W}}(\mathcal{P}^W_{\text{est}})|/|\mathsf{W}(\mathcal{P}^W_{\text{gt}})|$ for each node pair.

From the results shown in Fig. 10, we observe that shortest path quality has a similar behavior as absolute error, where PAINT has the best performance among all algorithms using uncontrollable routes. Compared to MAE results, PAINT yields an even larger gain in terms of SPRD, specifically 2.5x to 4x gain with NLANR-AMP dataset and 2x to 4x gain with RTC-CLOUD dataset. We can conclude that PAINT simultaneously achieves near ideal MAE and SP quality, which can be attributed to the additional regularization from SPE. In contrast, other algorithms do not factor in the relative difference between link weight estimations and so they can lead to SP changes due to single estimation error. Note that unsurprisingly FCR also has better performance of SPRD due to its high accuracy on majority of links and bound estimation for preserving relative delay but recall that this is meant to be an ideal but impractical baseline.

Fig. 11 takes a closer look at the SP estimation error for each individual node pair. The first observation is that the tail (95%) error of PAINT is at the same level of baselines average, confirming a consistent gain in terms of SPRD for any given node pair. The overall trend across different topologies is degraded compared to absolute link estimation error since link level errors accumulate when estimating SPs. The long tail is caused by the underestimation of links not on true SP and overestimation of links on true SP, resulting in estimated SPs deviating to higher delay routes. For PAINT, some node pairs may not have a better reference to validate against, especially in larger topologies leading to more error for paths between such node pairs.

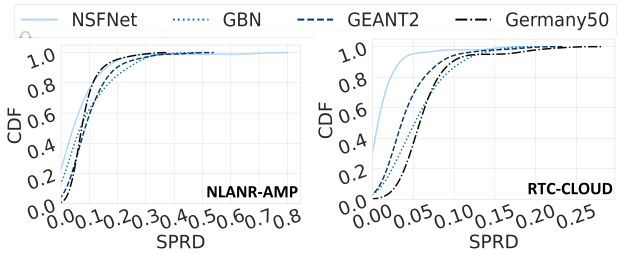
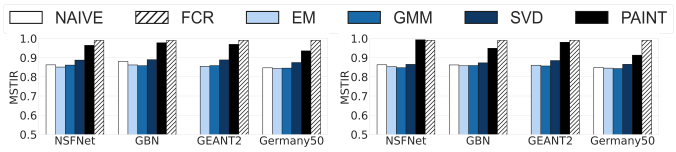


Fig. 11: CDF of SPRD across links with PAINT.



(a) NLANR-AMP (b) RTC-CLOUD

Fig. 12: Mean MSTIR of algorithms for 4 different topologies.

F. Minimal Spanning Tree Quality

Here we show that our PAINT algorithm featuring SP based regularization can also yield improvements for other downstream use cases besides SP by considering MST estimation. Note that MST is a subroutine for several problems, such as travelling salesman problem and maxflow-mincut problem, that are widely applied to improve cost-efficiency for networks [40], [41]. Spanning tree problem is also widely applied for constructing multicast tree for applications with high performance requirements [42], [43]. The correct classification of links plays an important role in MST for applications that need cost optimization and extracting the optimal sub network topology (e.g., [27]).

We use MST Identification Rate (MSTIR) as the metric to evaluate the accuracy for correct categorization of MST links, indicating the quality of graph cuts. We evaluate how a set of links with minimal weights are captured comparing to true MST. The accuracy alone is not sufficient to conclude that the quality of MST meets the need of applications, as mis-classification may lead to huge difference in delay performance. So we additionally consider MST Relative Difference (MSTRD) to calculate the relative difference between total weights of estimated MST and true MST. To compute these metrics, MST is generated from estimated link delays and compared against true MST on the same graph but with ground truth link weights.

In Fig. 12, we first examine the MST classification accuracy with different algorithms. PAINT has over 90% classification accuracy and improves by at least 6% and upto 9% compared to the best baseline. Such high accuracy can be attributed to link labeling and tuning, since MST contains with a close approximation the set of links with minimal delay. Although classification from all algorithms contain outliers, the least accurate outlier of PAINT is still better than the rest of the baselines with uncontrollable measurements (i.e., all except FCR), demonstrating the robust performance of PAINT. The fact that Naïve also has relatively good classification accuracy shows that values bounds from SLE can reflect the relative weight difference between links.

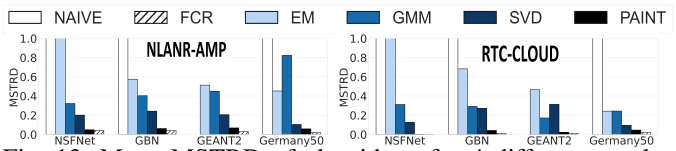


Fig. 13: Mean MSTRD of algorithms for 4 different topologies.

We make further comparison with respect to the relative error of estimated weight of MST (MSTRD). As shown in Fig. 13, PAINT provides 2x to 3x improvement with NLANR-AMP dataset and 2x to 5x improvement with RTC-CLOUD dataset. The performance of PAINT is within 2% of FCR, latter making the impractical assumption of full controllability of path measurements. Together with Fig. 12, these results show that PAINT yields an accurate approximation to resolve MST related problem in networks. Since statistical baselines do not have any regularization on the numerical order of link delay, the relative error and classification error of EM and GMM is close to Naïve. SVD has a relatively more accurate estimation since it performs a linear projection between lower dimensional space and the solution space, during which the principle components (the top K most frequently appeared links in the observation) can be estimated with high accuracy.

V. Conclusions

In this paper, we have studied the problem of inferring link metrics from a limited set of end-to-end path measurements under the framework of network tomography. We examine the problem from a path selection perspective and bring insights from shortest path estimation to accurate estimation of link metrics. Concretely, we have proposed PAINT, an online iterative algorithm that utilizes shortest path estimations to inform the estimation and refinement of link performance metrics. Evaluations using real-world topologies and measurement datasets focusing on link delays show that PAINT achieves 1.5x to 3x reduction in link delay estimation error relative to existing approaches while performing similarly to an impractical approach that needs fully controllable routes. In addition, PAINT achieves 2x to 5x improvement on shortest path and minimal spanning tree decision quality.

Recall that PAINT is designed to overcome the limitations of prior network tomography methods concerning the stability, controllability and visibility requirements. While our evaluations using real-world latency measurement datasets demonstrate that PAINT yields link metric estimates faster than the observed link quality fluctuations, further validation through real-world deployment is needed. Moreover, even though the PAINT design relaxes the requirement for monitor and path controllability, further evaluations modeling limited control are needed to demonstrate this capability. Finally, knowledge of link metric distribution or topology (i.e., visibility) is not required for PAINT but highlighting this feature through evaluations is an issue for future work. Note that lack of visibility manifests as greater link instability and reduced monitor/path control, both of which PAINT can cope with.

REFERENCES

[1] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, Í. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng, "Engineering egress with edge fabric: Steering oceans of content to the world," in *SIGCOMM*. ACM, 2017, pp. 418–431.

[2] K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. J. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain, V. Lin, C. Rice, B. Rogan, A. Singh, B. Tanaka, M. Verma, P. Sood, M. M. B. Tariq, M. Tierney, D. Trumic, V. Valancius, C. Ying, M. Kallahalla, B. Koley, and A. Vahdat, "Taking the edge off with Espresso: Scale, reliability and programmability for global internet peering," in *SIGCOMM*. ACM, 2017, pp. 432–445.

[3] Agora Inc., "Agora's software defined real-time network™ delivers real-time internet advantages over content deliver network," https://www.agora.io/en/wp-content/uploads/2021/02/Agora_WP_SD-RTN-Delivers-RealTime-Internet-Advantages.pdf, 2021.

[4] M. Calder, R. Gao, M. Schröder, R. Stewart, J. Padhye, R. Mahajan, G. Ananthanarayanan, and E. Katz-Bassett, "Odin: Microsoft's scalable fault-tolerant CDN measurement system," in *NSDI*. USENIX Association, 2018, pp. 501–517.

[5] B. Grozev, L. Marinov, V. Singh, and E. Ivov, "Last N: relevance-based selectivity for forwarding video in multimedia conferences," in *NOSSDAV*. ACM, 2015, pp. 19–24.

[6] B. García, L. López-Fernández, M. Gallego, and F. Gortázar, "Kurento: The swiss army knife of WebRTC media servers," *IEEE Commun. Stand. Mag.*, vol. 1, no. 2, pp. 44–51, 2017.

[7] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of the American statistical association*, vol. 91, no. 433, pp. 365–377, 1996.

[8] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statistical science*, pp. 499–517, 2004.

[9] E. Lawrence, G. Michailidis, V. N. Nair, and B. Xi, "Network tomography: A review and recent developments," in *Frontiers in statistics*. World Scientific, 2006, pp. 345–366.

[10] T. Bu, N. G. Duffield, F. L. Presti, and D. F. Towsley, "Network tomography on general topologies," in *SIGMETRICS*. ACM, 2002, pp. 21–30.

[11] A. Chen, J. Cao, and T. Bu, "Network tomography: Identifiability and fourier domain estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 12, pp. 6029–6039, 2010.

[12] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient identification of additive link metrics via network tomography," in *ICDCS*. IEEE Computer Society, 2013, pp. 581–590.

[13] H. Li, Y. Gao, W. Dong, and C. Chen, "Bound-based network tomography for inferring interesting link metrics," in *INFOCOM*. IEEE, 2020, pp. 1588–1597.

[14] H. H. Song, L. Qiu, and Y. Zhang, "NetQuest: a flexible framework for large-scale network measurement," *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, pp. 106–119, 2009.

[15] A. Orda and R. Rom, "Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length," *J. ACM*, vol. 37, no. 3, pp. 607–625, 1990.

[16] C. Feng, L. Wang, K. Wu, and J. Wang, "Bound inference in network performance tomography with additive metrics," *IEEE/ACM Trans. Netw.*, vol. 28, no. 4, pp. 1859–1871, 2020.

[17] R. Diestel, *Graph Theory*. Springer-Verlag Berlin Heidelberg, 2005, vol. 173.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 3rd ed. MIT press, 2009, ch. 6, pp. 615–623.

[19] D. Ghita, K. J. Argyraki, and P. Thiran, "Network tomography on correlated links," in *Internet Measurement Conference*. ACM, 2010, pp. 225–238.

[20] S. Keshav, *Mathematical Foundations of Computer Networking*. Addison-Wesley, 2012, ch. 3.

[21] G. Liang and B. Yu, "Maximum pseudo likelihood estimation in network tomography," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2043–2053, 2003.

[22] K. Deng, Y. Li, W. Zhu, Z. Geng, and J. S. Liu, "On delay tomography: Fast algorithms and spatially dependent models," *IEEE Trans. Signal Process.*, vol. 60, no. 11, pp. 5685–5697, 2012.

[23] Y. Chen, D. Bindel, and R. H. Katz, "Tomography-based overlay network monitoring," in *Internet Measurement Conference*. ACM, 2003, pp. 216–231.

[24] D. B. Chua, E. D. Kolaczyk, and M. Crovella, "Efficient monitoring of end-to-end network properties," in *INFOCOM*. IEEE, 2005, pp. 1701–1711.

[25] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 1092–1103, 2006.

[26] A. Gopalan and S. Ramasubramanian, "On identifying additive link metrics using linearly independent cycles and paths," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 906–916, 2012.

[27] S. Legtchenko, N. Chen, D. Cletheroe, A. Rowstron, H. Williams, and X. Zhao, "XFabric: A reconfigurable In-Rack network for Rack-Scale computers," in *NSDI'16*. Santa Clara, CA: USENIX Association, Mar. 2016, pp. 15–29.

[28] C. Gray, C. Mosig, R. Bush, C. Pelsser, M. Roughan, T. C. Schmidt, and M. Wählisch, "BGP beacons, network tomography, and bayesian computation to locate route flap damping," in *IMC '20: ACM Internet Measurement Conference, Virtual Event, USA, October 27-29, 2020*. ACM, 2020, pp. 492–505. [Online]. Available: https://doi.org/10.1145/3419394.3423624

[29] Y. Jin, S. Renganathan, G. Ananthanarayanan, J. Jiang, V. N. Padmanabhan, M. Schröder, M. Calder, and A. Krishnamurthy, "Zooming in on wide-area latencies to a global cloud provider," in *Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM 2019, Beijing, China, August 19-23, 2019*, J. Wu and W. Hall, Eds. ACM, 2019, pp. 104–116. [Online]. Available: https://doi.org/10.1145/3341302.3342073

[30] K. Chen, C. Huang, P. Huang, and C. Lei, "Quantifying skype user satisfaction," in *SIGCOMM*. ACM, 2006, pp. 399–410.

[31] N. Q. M. Khiem, G. Ravindra, and W. T. Ooi, "Towards understanding user tolerance to network latency in zoomable video streaming," in *ACM Multimedia*. ACM, 2011, pp. 977–980.

[32] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear algebra*. Springer, 1971, pp. 134–151.

[33] K. Kuttler, *Linear Algebra, Theory And Applications*. The Saylor Foundation, 2012.

[34] S. Knight, H. X. Nguyen, N. Falkner, R. A. Bowden, and M. Roughan, "The internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, 2011. [Online]. Available: https://doi.org/10.1109/JSAC.2011.111002

[35] J. Suárez-Varela, A. Mestres, J. Yu, L. Kuang, H. Feng, P. Barlet-Ros, and A. Cabellos-Aparicio, "Feature engineering for deep reinforcement learning based routing," in *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*. IEEE, 2019, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ICC.2019.8761276

[36] T. McGregor, H. Braun, and J. Brown, "The NLANR network analysis infrastructure," *IEEE Commun. Mag.*, vol. 38, no. 5, pp. 122–128, 2000. [Online]. Available: https://labs.ripe.net/datarepository/data-sets/nlanr-amp-data/

[37] J. Jiang, S. Sun, V. Sekar, and H. Zhang, "Pytheas: Enabling data-driven quality of experience optimization using group-based exploration-exploitation," in *NSDI*. USENIX Association, 2017, pp. 393–406.

[38] J. Jiang, R. Das, G. Ananthanarayanan, P. A. Chou, V. N. Padmanabhan, V. Sekar, E. Dominique, M. Goliszewski, D. Kukoleca, R. Vafin, and H. Zhang, "VIA: Improving internet telephony call quality using predictive relay selection," in *SIGCOMM*. ACM, 2016, pp. 286–299.

[39] R. Gouareb, V. Friderikos, and A. Aghvami, "Virtual network functions routing and placement for edge cloud latency minimization," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2346–2357, 2018.

[40] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees," *Oper. Res.*, vol. 18, no. 6, pp. 1138–1162, 1970.

[41] L. Massoulié, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," in *INFOCOM*. IEEE, 2007, pp. 1073–1081.

[42] A. Young, J. Chen, Z. Ma, A. Krishnamurthy, L. L. Peterson, and R. Wang, "Overlay mesh construction using interleaved spanning trees," in *Proceedings IEEE INFOCOM 2004, The 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, Hong Kong, China, March 7-11, 2004*. IEEE, 2004. [Online]. Available: https://doi.org/10.1109/INFCOM.2004.1354512

[43] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, "OMNI: an efficient overlay multicast infrastructure for real-time applications," *Comput. Networks*, vol. 50, no. 6, pp. 826–841, 2006. [Online]. Available: https://doi.org/10.1016/j.comnet.2005.07.023