FRANTIC: A Fast Reference-based Algorithm for Network Tomography via Compressive Sensing

Sheng Cai Mayank Bakshi Sidharth Jaggi Minghua Chen

The Chinese University of Hong Kong

Abstract

We study the problem of link and node delay estimation in undirected networks when at most k out of n links or nodes in the network are congested. Our approach relies on end-to-end measurements of path delays across pre-specified paths in the network. We present a class of algorithms that we call FRANTIC [1]. The FRANTIC algorithms are motivated by compressive sensing; however, unlike traditional compressive sensing, the measurement design here is constrained by the network topology and the matrix entries are constrained to be positive integers. A key component of our design is a new compressive sensing algorithm SHO-FA-INT that is related to the SHO-FA algorithm [1] for compressive sensing, but unlike SHO-FA, the matrix entries here are drawn from the set of integers $\{0, 1, \ldots, M\}$. We show that $\mathcal{O}(k \log n / \log M)$ measurements suffice both for SHO-FA-INT and FRANTIC . Further, we show that the computational complexity of decoding is also $\mathcal{O}(k \log n / \log M)$ for each of these algorithms. Finally, we look at efficient constructions of the measurement operations through Steiner Trees.

I. INTRODUCTION

Monitoring performance characteristics of individual links is important for diagnosing and optimizing network performance. Making direct measurements for each link, however, is impractical in large-scale networks because (i) nodes inside the networks may not be available to

[1] FRANTIC stands for **F**ast **R**eference-based **A**lgorithm for **N**etwork **T**omography v**I**a **C**ompressive sensing.

carry out measurements due to physical or protocol constraints, and (ii) measuring each link *separately* incurs excessive control-traffic overhead and is not scalable.

A viable alternative approach is network tomography [2]. It aims to infer the performance characteristics of internal links by *path measurements* between controllable nodes, where a path measurement is function of the characteristics of the links on the path. It does not require access to all the nodes and, more importantly, it allows clever solutions to leverage the network structure (*e.g.*, topology and graph properties) to *jointly* infer the performance characteristics of multiple links via path measurements. Many existing work have explored such insight to design excellent solutions that are able to infer the congested links with much less measurements than the direct link measurement approach [3]–[7]. See [8] for a survey.

Recently, Xu *et al.* [9] further argue that usually only a small fraction of network links, say k out of total $|\mathcal{E}|$ links ($k \ll |\mathcal{E}|$), are congested (*i.e.*, experiencing large congestion delay or high packet loss rate). They interpret each path measurement as a linear combination of the delays or loss rates of the k congested links. With these understanding, the problem of network tomography can be viewed as recovering a k-sparse link vector from a set of linear measurements.

Exploiting the "sparse congestion structure" insight, Xu *et al.* [9] propose a compressive sensing based scheme that can identify any k congested links using $\mathcal{O}(T_{\mathcal{N}} k \log |\mathcal{E}|)$ path measurements over any sufficiently-connected graph. Here, each of the path measurement is a random walk on the graph, and $T_{\mathcal{N}}$ is the mixing time of the random walk. Further, they show that one can actually *recover* the performance characteristics of any k congested links with again $\mathcal{O}(T_{\mathcal{N}} k \log |\mathcal{E}|)$ path measurements by using ℓ_1-minimization. Similar results are also obtained by [10]–[12]. Given all these exciting results, a natural question is that can we do better and how?

II. Our contribution

A. Summary

In this paper, we build upon our recently developed compressive sensing algorithm named SHO-FA [1] to design a new network tomography solution that we call FRANTIC . FRANTIC

achieves the following performance:

- FRANTIC can identify any k congested links (or nodes) out of n and recover the corresponding link (or node) performance characteristics using $\mathcal{O}(\rho k \log n / \log M)$ path measurements with a high probability. Here, $M \in \mathbb{N}$ and $\rho \in \Omega(1) \cap o(n/k)$ are design parameters. See Section II-C for a discussion.

- The FRANTIC decoding algorithm can recover the link (or node) performance characteristics in $\mathcal{O}(\rho k \log n / \log M)$ steps.

As compared to the solution in [9], our solution improves both the number of measurements and the number of recovery steps from $\mathcal{O}(T_{\mathcal{N}} k \log n)$ to $\mathcal{O}(k)$ (obtainable by setting $M = \mathcal{O}(n)$).

B. Techniques and results

The main techniques that lead to these improvements are as follows. First, in Section VI, we develop an efficient compressive sensing algorithm SHO-FA-INT when the entries of the measurement matrix are constrained to be positive integers. Our algorithm borrows key ideas from a prior work [1] that studies compressive sensing in the unconstrained setting. A key technique here is to group together measurements and choose the "weights" of the measurement matrix as co-prime vectors. This ensures that each network link has a distinct signature in the measurement output, which allows us to decode the delay values for congested links in an iterative manner. Theorem 1 states the performance guarantees of our algorithm. Next, we propose a design for measuring the delay on congested links in a network in Section VII. An important insight in our design is that by using local loops at individual edges, end-to-end delay measurements can be designed to assign different integer weights to delays for different edges. We start with a compressive sensing matrix given by SHO-FA-INT and emulate the output of the matrix by first designing two correlated network paths, and then cancelling out the contribution of unwanted links by subtracting one from another. Theorems 2-4 state the performance guarantees of the FRANTIC algorithms. We also note that the path lengths required for FRANTIC can be suitably optimised by using Steiner Trees and network decomposition. Theorem 4 and subsequent

discussions point this out.

C. Explanation of design parameters

The parameter M denotes the maximum number of times a test packet may travel over any edge. In many present day networks, the value of M is usually fixed to be a small constant. In this setting, our algorithm requires $\mathcal{O}(k \log n)$ measurements and decoding steps. Additionally, if M is allowed to increase with the network size (possibly, in future generation networks), the number of measurements and decoding complexity our algorithms may be decreased to $\mathcal{O}(k)$.

The parameter ρ is a design parameter that controls the tradeoff between the measurement path lengths and the number of measurements required. When $\rho = 1$, we require $\mathcal{O}(k \log n / \log M)$ measurements with path lengths $\mathcal{O}(nD/k)$. On the other extreme, if ρ is set to be $n/(k\omega(1))$, we require upto $o(n)$ measurements but with as little as $\omega(D)$ path length. In our exposition, we prove the correctness of our schemes for the case when $\rho = 1$. The results for other values of ρ follow from this analysis by pretending that the network has ρk congested nodes instead of k.

III. Model and problem formulation

Network and delay model: Let $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ be a undirected network with node set $\mathcal{V}$ and link set $\mathcal{E}$. In this paper, we consider the reference-based tomography problem where the topology of $\mathcal{N}$ is known. We assume that $\mathcal{N}$ is connected. We say that a node $v \in \mathcal{V}$ has delay d_v if every packet that passes through v is delayed by d_v. Similarly, a link $e \in \mathcal{E}$ has delay d_e if every packet passing through e in any direction is delayed by d_e. We say a node or link is *congested* if the delay associated with it is non-zero. A congested node is called *isolated* if there exists one of its neighbours which is not congested. Let $\boldsymbol{d}_{\mathcal{V}} = (d_v : v \in \mathcal{V})$ and $\boldsymbol{d}_{\mathcal{E}} = (d_e : e \in \mathcal{E})$. Both $\boldsymbol{d}_{\mathcal{V}}$ and $\boldsymbol{d}_{\mathcal{E}}$ are unknown but have at most k non-zero coordinates.

Measurement model: Each measurement is performed by sending test packets over pre-determined paths[2] and measuring the end-to-end time taken for its transmission. Some nodes (resp. links) may be visited more than once in a given path. As a result, each measurement output y_i,

[2]In present day networks, this may be accomplished by employing source-based routing (c.f. [16]) for the test packets.

Reference	Type	# Measurements	Decoding Complexity	Path Length	Network Topology
[11]	Node	$R\mathcal{O}(k\log(\lvert\mathcal{V}\rvert/k))+R+1$	cs with $0-1$ matrix	–	General graph, R is the radius of the graph
	Node	$\mathcal{O}(rk\log(\lvert\mathcal{V}\rvert/k))+r$	cs with $0-1$ matrix	–	If G has an r-partition
	Node	$\mathcal{O}(2k\log(\lvert\mathcal{V}\rvert/2k))+r$	cs with $0-1$ matrix	–	Erdos-Renyi random graph $G(\lvert\mathcal{V}\rvert,p)$, with $p=\beta\log\lvert\mathcal{V}\rvert/\lvert\mathcal{V}\rvert$ and $\beta\geq 2$
[9]	Edge	$\mathcal{O}(T_{\mathcal{N}}k\log\lvert\mathcal{E}\rvert)$	l_1 minimization	$\mathcal{O}(\lvert E\rvert/k)$	G is a (D,c)-uniform graph, $D\geq D_0=\mathcal{O}(c^2kT_{\mathcal{N}}^2)$.
[10]	Edge	$\mathcal{O}(k\log(\lvert\mathcal{E}\rvert/k))$	l_1 minimization	–	Network is 1-identifiable
[12]	Node	$\mathcal{O}\left(c^4k^2T_{\mathcal{N}}^2\log(\lvert\mathcal{V}\rvert/d)\right)$	Disjunct matrix	$\mathcal{O}(\lvert\mathcal{V}\rvert/(c^3kT_{\mathcal{N}}))$	G is a (D,c)-uniform graph. $D\geq D_0=\mathcal{O}(c^2kT_{\mathcal{N}}^2)$.
	Edge	$\mathcal{O}(c^4k^2T_{\mathcal{N}}^2\log(\lvert\mathcal{E}\rvert/d))$	Disjunct matrix	$\mathcal{O}(\lvert\mathcal{V}\rvert D/(c^3kT_{\mathcal{N}}))$	
	Node	$\mathcal{O}(c^8k^3T_{\mathcal{N}}^4\log(\lvert\mathcal{V}\rvert/d))$	Disjunct matrix	unbounded (sink node)	
	Edge	$\mathcal{O}(c^9k^3DT_{\mathcal{N}}^4\log(\lvert\mathcal{E}\rvert/d))$	Disjunct matrix	unbounded (sink node)	
	Node	$\mathcal{O}(k^2(\log^3\lvert\mathcal{V}\rvert))/(1-p)^2$	Disjunct matrix	$\mathcal{O}(\lvert\mathcal{V}\rvert/(c^3kT_{\mathcal{N}}))$	G is D-regular expander graph or Erdos-Renyi random graph, $G(\lvert\mathcal{V}\rvert,D/\lvert\mathcal{V}\rvert)$, with $D\geq D_0=\Omega(k\log^2\lvert\mathcal{V}\rvert)$.
	Edge	$\mathcal{O}(k^2(\log^3\lvert\mathcal{E}\rvert))/(1-p)^2$	Disjunct matrix	$\mathcal{O}(\lvert\mathcal{V}\rvert D/(c^3kT_{\mathcal{N}}))$	
	Node	$\mathcal{O}(k^3(\log^5\lvert\mathcal{V}\rvert))/(1-p)^2$	Disjunct matrix	unbounded (sink node)	
	Edge	$\mathcal{O}(k^3D(\log^5\lvert\mathcal{E}\rvert))/(1-p)^2$	Disjunct matrix	unbounded (sink node)	
This paper	Node	$\mathcal{O}(k\log\lvert\mathcal{V}\rvert/\log M)$	$\mathcal{O}(k\log\lvert\mathcal{V}\rvert/\log M)$	$\mathcal{O}(D\lvert\mathcal{V}\rvert/k)$	General Graph D is the diameter of the graph
	Edge	$\mathcal{O}(k\log\lvert\mathcal{E}\rvert/\log M)$	$\mathcal{O}(k\log\lvert\mathcal{E}\rvert/\log M)$	$\mathcal{O}(D\lvert\mathcal{E}\rvert/k)$	

Table I

Partial literature review: [11] considers the node delay estimation problem where a set of nodes can be measured together in one measurement if and only if the induced subgraph is connected and each measurement is an additive sum of values at the corresponding nodes. The generated sensing matrix is a $0-1$ matrix, therefore the decoding complexity mainly depends on which binary compressive sensing algorithm is used. General graph and some special graphs are studied. The idea of a binary compressive sensing algorithm is borrowed by [10] where a single edge delay estimation problem is studied and the estimation is done using l_1 minimization. In [9], a random-walk based approach is proposed to solve the k-edge delay estimation problem. $T_{\mathcal{N}}$ is the $\frac{1}{(2c|\mathcal{V}|)^2}$-mixing time of $\mathcal{N}$. The networks with degree-bounded assumption are studied. Similar to [9], [12] uses random-walk measurements to solve both node and edge failure localization problem where group testing (non-linear version of compressive sensing) algorithm is used. The goal is to generate disjunct matrices which are suitable for group testing. The start points of measurements can be chosen within a fixed set of designated vertices, or, chosen randomly among all vertices of the graph. The first type of construction which don't have the length bound covers the case that only a small subset of vertices are accessible as the starting points of the measurements. Separately, the problem of edge failure localization has also been studied in the optical networking literature [13]–[15]. [13], which consider the single edge failure localization, has the same flavor as [11]. Binary-search type algorithms are proposed for some special graphs. For the general graphs, the upper bound on the number of measurements required for single edge failure localization is $\mathcal{O}(D(\mathcal{N})+\log^2(|\mathcal{V}|))$ where $D(\mathcal{N})$ is the diameter of the graph. In [14], the problem of multi-link failure localization is considered. For small networks, tree-decomposition based method has the upper bound on the number of trials is $\min(\mathcal{O}(D(\mathcal{N})\log|\mathcal{V}|),\mathcal{O}(D(\mathcal{N})+\log^2(|\mathcal{V}|)))$. For the large-scale networks, random-walk based method similarly to [12] is proposed. They also consider the practical constraints such as the number of failed links cannot be known beforehand. In [15], the solution proposed is based on the $(k+2)$-edge-connected network for k link failures localization.

$i=1,2,\ldots,m$, is a weighted sum of delays involving nodes and links that lie in the measurement path, where, weight of a given node or link is the number of times it is visited by the measurement path. In this paper, we consider two kinds of measurements – *node measurements* and *link measurements*. In the node (resp. link) measurements, we associate each node (resp. link) with a real-valued delay and the objective is to reconstruct the node (resp. link) delay vector $\boldsymbol{d}_{\mathcal{V}}$ (resp. $\boldsymbol{d}_{\mathcal{E}}$) given the collection of measurement outputs.

1. <u>*Node measurements*</u>*:* In the node measurement model, we associate each node with a real valued delay (see [11], for example). Let $\mathcal{S}\subseteq\mathcal{V}$ denote a subset of nodes in $\mathcal{N}$. Let $\mathcal{E}_{\mathcal{S}}$ denote the subset of links with both ends in $\mathcal{S}$, then $\mathcal{N}_{\mathcal{S}}=(\mathcal{S},\mathcal{E}_{\mathcal{S}})$ is the induced subgraph of $\mathcal{N}$. A set $\mathcal{S}$ of nodes can be measured together in one measurement if and only if $\mathcal{N}_{\mathcal{S}}$ is connected.

n	Total number of links (or nodes) in the network
k	Number of congested links (or nodes) in the network
M	The maximum number of times a test packet may travel over any edge
D	The diameter of $\mathcal{N}$
$T_{\mathcal{N}}$	The mixing time of the random walk over graph $\mathcal{N}$
ρ	A design parameter that controls the tradeoff between the path lengths and the number of the measurements
$\mathcal{N}$	$\mathcal{N} = (\mathcal{V}, \mathcal{E})$, a undirected network with node set $\mathcal{V}$ and link set $\mathcal{E}$
d_v	Time taken by a test packet to pass through node $v \in \mathcal{V}$
$\boldsymbol{d}_{\mathcal{V}}$	Node delay vector of length $\lvert\mathcal{V}\rvert$
d_e	Time taken by a test packet to pass through link $e \in \mathcal{E}$ in any direction
$\boldsymbol{d}_{\mathcal{E}}$	Link delay vector of length $\lvert\mathcal{E}\rvert$

II-A. **Network Parameters**

R	$R \in \mathbf{N}^+$ such that $M^R/\zeta(R) \geq 3n$ where $\zeta(\cdot)$ be the Riemann zeta function
$\boldsymbol{y}$	Measurement output of length $m = R\mu$
$\mathbf{A}$	Measurement matrix of dimension $R\mu \times n$
$a_{ij}^{(r)}$	The r-th row entry in the j-th column of the i-th group of $\mathbf{A}$ for $r = 1, 2, \ldots, R$, $i = 1, 2, \ldots, \mu$ and $j = 1, 2, \ldots, n$
$\mathcal{G}_{n,\mu}$	A bipartite graph with left vertex set $\{1, 2, \ldots, n\}$ and right vertex set $\{1, 2, \ldots \mu\}$
$N(S)$	The set of right neighbours of a subset of left nodes S in $\mathcal{G}_{n,\mu}$
$\boldsymbol{P}$	A path of length T over the network $\mathcal{N} = (\mathcal{V}, \mathcal{E})$, *i.e.,* a sequence $(e_1, e_2, \ldots, e_T)$ of links from $\mathcal{E}$
$W(\boldsymbol{P}, e)$	The multiplicity of a link $e \in \mathcal{E}$ given a path $\boldsymbol{P}$, *i.e.,* the number of times $\boldsymbol{P}$ visits e
$\Delta(\boldsymbol{P})$	The end-to-end delay for a path $\boldsymbol{P}$

II-B. **Design Variables**

Table II
TABLE OF NOTATIONS

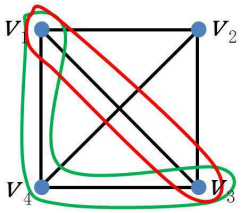
2. *Link measurements:* In link measurement setup, we associate each link with a real valued delay. Let $\mathcal{T} \subseteq \mathcal{E}$ denote a subset of links in $\mathcal{N}$. A set $\mathcal{T}$ of links can be measured together in one measurement if and only if there exists a path that traversed each link in $\mathcal{T}$.

For each of these models, we express the measurement output as a vector $\boldsymbol{y} \in \mathbb{R}^m$ that is related to the delay vector through a measurement matrix $\mathbf{A}$ through matrix multiplication.

IV. KEY IDEAS

In this section, we present some key observations and challenges that this paper focuses on. We begin with the observation that there is a high-level connection between the compressive sensing and the network tomography problem. As noted in the previous section, network tomography can be treated as a problem of solving a system of linear equations. Under the assumption that the underlying unknown vector is sparse, it is natural to think of it as a compressive sensing problem [17] [18] [19]. Building on this intuition, network tomography can be formulated as

the following compressive sensing problem: i) design a matrix $\mathbf{A}$, ii) obtain delay measurements $\boldsymbol{y} = \mathbf{A}\boldsymbol{d}_{\mathcal{V}}$ iii) reconstruct $\boldsymbol{d}_{\mathcal{V}}$ from $\boldsymbol{y}$. Fig. 1 illustrates this connection in a complete graph. Since each subset of nodes in a complete graph induces a connected subgraph, we can freely choose the locations of non-zero entries in each row of $\mathbf{A}$. Then, any compressive sensing algorithm with 0-1 measurement matrix [20] [21] can be applied to recover the vector $\boldsymbol{d}_{\mathcal{V}}$. However, when we go



$$\boldsymbol{y} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \boldsymbol{d}_{\mathcal{V}}$$

Figure 1. Node Delay Estimation: For a complete graph with four vertices. We can get any measurements we want since each subgraph of a complete graph is connected. For example, the subgraphs induced by $\{v_1, v_3\}$ (covered by red cycle) and $\{v_1, v_3, v_4\}$ (covered by green cycle) are connected, therefore we get the measurements $[1\ 0\ 1\ 0]\boldsymbol{d}_{\mathcal{V}}$ and $[1\ 0\ 1\ 1]\boldsymbol{d}_{\mathcal{V}}$ respectively.

beyond complete graphs and node measurements, it is not straightforward to apply compressive sensing directly. The network topology may impose constraints on implementable measurement matrices (See Figs. 2,3,5).

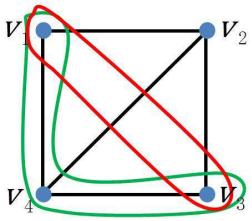


Figure 2. General Networks: If the link (v_1, v_3) is removed from the original complete graph, we cannot get the measurement $[1\ 0\ 1\ 0]\boldsymbol{d}_{\mathcal{V}}$ any longer.

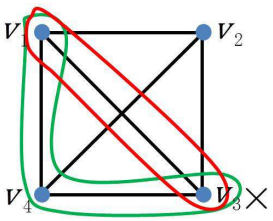


Figure 3. Inaccessible Nodes: If there is some constraint that we can not access to v_3 and v_3 is the destination of the paths for us to get the measurement, then any measurement in Fig. 1 is not available.

Xu *et al*. [9] get around some of these problems by using random walks. One drawback of their approach is that it involves a factor of mixing time $T_{\mathcal{N}}$ for both the number of measurements and path length. For networks without sufficient connectivity, mixing time may be very large, *e.g.*, cycle graph, $T_{\mathcal{N}} = \mathcal{O}(|\mathcal{E}|^2)$. In the following, we propose two news ideas that enable us to circumvent the above problem.

Idea 1: *Cancellation enables selecting disconnected subsets of links and nodes.* The idea here is similar to that used in [11] where they use the structure called hub to get arbitrary measurement matrix. However, they only consider the node delay model and special graphs which have r-partition. In this paper, we expand this approach to both link delay and node delay models. By considering correlated measurements, we can cancel out the contribution of the undesired links and nodes in a given measurement. Using this approach, we can mimic arbitrary measurements on general graphs. See Fig. 4 as an illustration. One drawback of the cancellation based approach is that if the selected measurement has too many disjoint components, then the number of measurements required is very large. In Fig. 4, the number of cancellations is 2.

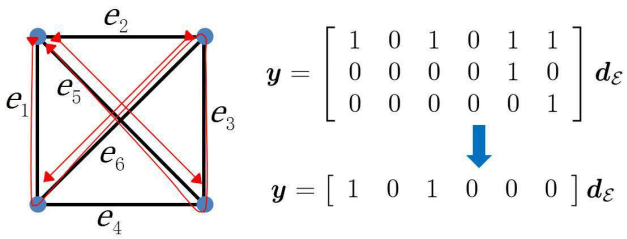


Figure 4. Cancellation: There are three paths in this graph:$\{e_1e_6e_3e_5\}$, $\{e_5\}$ and $\{e_6\}$. Triangles indicate the source and destination of a path. Correspondingly, we can derive three measurements $[1\ 0\ 1\ 0\ 1\ 1]\boldsymbol{d}_{\mathcal{E}}$, $[0\ 0\ 0\ 0\ 1\ 0]\boldsymbol{d}_{\mathcal{E}}$, and $[0\ 0\ 0\ 0\ 1]\boldsymbol{d}_{\mathcal{E}}$. Subtracting the second and the third measurements from the first measurement, we get the measurement $[1\ 0\ 1\ 0\ 0\ 0]\boldsymbol{d}_{\mathcal{E}}$ which cannot be got by just one path.

Idea 2: *Weighted measurements reduce the number of cancellations required and allow us to implement arbitrary integer valued matrices.* The insight here is that if we have two paths along the same set of links, we can assign different weights for each link (or node) on these paths by performing local loops. Specifically, for a given set of weights on a subset of links (or nodes), we construct two measurements - a spanning measurement, and a weighted measurement. The spanning measurement is constructed by finding any path that visits through all the links (or nodes) in the desired subset at least once. The weighted measurement, then follows the same set of edges as the spanning measurement, but visits each link (or node) an additional number of times in accordance with the desired weight for that link (or node). Finally, we subtract the end-to-end delay for the weighted path from that of the spanning path to get an output that is proportional to the output of the corresponding compressive sensing problem (See Fig. 7). These

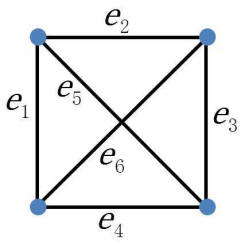


Figure 5. Edge Delay Estimation: We know that we can not get arbitrary measurement by one path even if the graph is complete. (*e.g.*, the measurement $[0\ 0\ 0\ 0\ 1\ 1]\boldsymbol{d}_{\mathcal{E}}$ cannot be got since there is no path just visiting e_5 and e_6.)

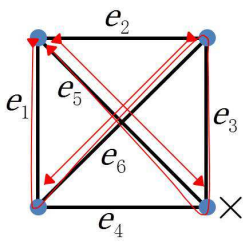


Figure 6. Inaccessible Nodes: The second measurement in Fig. 4 cannot be got since the v_3 which is the destination of the second path is not accessible (the same node identifier in Fig. 1).

ideas enable us to reduce the network tomography problem to a compressive sensing problem with integer valued matrices. In the Section VI, we present an efficient compressive sensing algorithm with integer entries.

V. Main Results

In this section, we state the main results of this paper. Let $\rho \in \Omega(1) \cap o(|\mathcal{E}|/k)$ be a design parameter.

Theorem 1 (Compressive sensing via integer matrices)**.** *Let $M \in \mathbb{Z}^+$. There exists a constant c such that whenever $m > ck\lceil \log n / \log M \rceil$, the ensemble of $\mathbb{Z}_M$-valued matrices $\{\mathbf{A}_{m\times n}\}$ designed in Section VI and the* SHO-FA-INT *reconstruction algorithm has the following properties:*

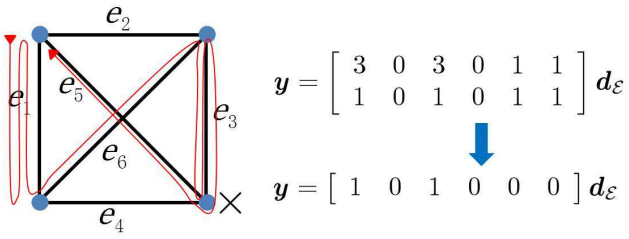


Figure 7. Cancellation using weighted measurements: To get the measurement $[1\ 0\ 1\ 0\ 0\ 0]\boldsymbol{d}_{\mathcal{E}}$, we design the paths as follows. First, we just follow the path $\{e_1e_6e_3e_5\}$, we get the measurement $[1\ 0\ 1\ 0\ 1\ 1]\boldsymbol{d}_{\mathcal{E}}$. Second, when visiting e_1 and e_3 for the first time, the probe does one more local loop for both links to get the measurement $[3\ 0\ 3\ 0\ 1\ 1]\boldsymbol{d}_{\mathcal{E}}$. Finally, we take the difference of these two measurements and divide the result by 2. Note that 1) Only one cancellation is required. 2) Even if v_3 is inaccessible, we still can achieve the two target measurements. 3) One additional local loops at e_1 in the second step (so that e_1 is visited 5 times), allows us the measurement $[2\ 0\ 1\ 0\ 0\ 0]\boldsymbol{d}_{\mathcal{E}}$. Thus, controlling the number of local loops allows us to implement other ensembles of measurement matrices.

1) *Given* $(\mathbf{A}_{m\times n}, \mathbf{A}_{m\times n}\boldsymbol{d})$ *as input, where* $\boldsymbol{d}$ *is an arbitrary* k*-sparse vector in* $\mathbb{R}^n$*,* SHO-FA-INT *outputs a vector* $\hat{\boldsymbol{d}} \in \mathbb{R}^n$ *that equals* $\boldsymbol{d}$ *with probability at least* $1 - \mathcal{O}(1/k)$ *under the distribution of* $\mathbf{A}_{m\times n}$ *over the ensemble* $\{\mathbf{A}_{m\times n}\}$.
2) *Given* $\mathbf{A}_{m\times n}\boldsymbol{d}$*,* $\hat{d}$ *is reconstructed in* $\mathcal{O}(k\lceil \log n/\log M\rceil)$ *arithmetic operations.*
3) *Each row of* $\mathbf{A}_{m\times n}$ *has* $\mathcal{O}(n/k)$ *non-zeros in expectation.*

Theorem 2 (Network tomography for link congestion)**.** *Let* $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ *be an undirected network of diameter* D *such that at most* k *have unknown non-zero link delays. Let* $M \in \mathbb{Z}^+$ *Then, the* FRANTIC *algorithm has the following properties:*

1) FRANTIC *requires* $\mathcal{O}(\rho k\lceil \log |\mathcal{E}|/\log M\rceil)$ *measurements.*
2) *For every edge delay vector* $\boldsymbol{d}_{\mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}$*,*FRANTIC *outputs* $\hat{\boldsymbol{d}}_{\mathcal{E}}$ *that equals* $\boldsymbol{d}_{\mathcal{E}}$ *with probability* $1 - \mathcal{O}(1/\rho k)$.
3) *The* FRANTIC *reconstruction algorithm requires* $\mathcal{O}(\rho k\lceil \log |\mathcal{E}|/\log M\rceil)$ *arithmetic operations.*
4) *The number of links of* $\mathcal{N}$ *traversed by each test measurement packet in* FRANTIC *is* $\mathcal{O}(D|\mathcal{E}|/\rho k)$ *and the total number of hops for each packet is* $\mathcal{O}(DM|\mathcal{E}|/\rho k)$.

Definition 1 (Isolated congested node)**.** *A congested node is called isolated if there exists one of its neighbours which is not congested.*

Theorem 3 (Network tomography for node congestion)**.** *Let* $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ *be an undirected network of diameter* D *such that at most* k *have unknown non-zero node delays and all congested nodes are isolated. Let* $M \in \mathbb{Z}^+$ *Then, the* FRANTIC *algorithm has the following properties:*

1) FRANTIC *requires* $\mathcal{O}(\rho k\lceil \log |\mathcal{V}|/\log M\rceil)$ *measurements.*
2) *For every edge delay vector* $\boldsymbol{d}_{\mathcal{V}} \in \mathbb{R}^{|\mathcal{V}|}$*,* FRANTIC *outputs* $\hat{\boldsymbol{d}}_{\mathcal{V}}$ *that equals* $\boldsymbol{d}_{\mathcal{V}}$ *with probability* $1 - \mathcal{O}(1/\rho k)$.
3) *The* FRANTIC *reconstruction algorithm requires* $\mathcal{O}(\rho k\lceil \log |\mathcal{V}|/\log M\rceil)$ *arithmetic operations.*
4) *The number of links of* $\mathcal{N}$ *traversed by each test measurement packet in* FRANTIC *is* $\mathcal{O}(D|\mathcal{V}|/\rho k)$ *and the total number of hops for each packet is* $\mathcal{O}(DM|\mathcal{V}|/\rho k)$.

VI. SHO-FA-INT ALGORITHM FOR COMPRESSIVE SENSING

We begin by describing a new compressive sensing algorithm SHO-FA-INT which has several properties that are desirable for our application. SHO-FA-INT is related to the SHO-FA algorithm – originally developed in the unconstrained compressive sensing setting [1] – but differs from it in that the non-zero entries of the sensing matrix $\mathbf{A}$ are constrained to be positive integers less than or equal to some $M \in \mathbb{N}$. [3]

Let $\{\mathcal{G}_{n,\mu}\}_{n,\mu\in\mathbb{N}}$ be an ensemble of left-regular bipartite graphs, where each $\mathcal{G}_{n,\mu}$ is a bipartite graph with left vertex set $\{1,2,\ldots,n\}$ and right vertex set $\{1,2,\ldots,\mu\}$. For each left vertex $j \in \{1,2,\ldots,n\}$, we pick three distinct vertices uniformly at random from the set of right vertices $\{1,2,\ldots,\mu\}$.

Measurement Design: Let $\zeta(\cdot)$ be the Riemann zeta function. Let $R \in \mathbb{N}^+$ such that $M^R/\zeta(R) \geq 3n$ and let $[M]$ denote the set $\{1,2,\ldots,M\}$. Given the graph $\mathcal{G}_{n,\mu}$, we design a $R\mu \times n$ measurement matrix $\mathbf{A}(= \mathbf{A}_{R\mu\times n})$ as follows. First, we partition the rows of $\mathbf{A}$ into μ groups of rows, each consisting of R consecutive rows as follows.

$$\mathbf{A} = \left[\begin{array}{c:c:c:c} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{11}^{(R)} & a_{12}^{(R)} & \ldots & a_{2n}^{(R)} \\ \hdashline a_{21}^{(1)} & a_{22}^{(1)} & \ldots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{21}^{(R)} & a_{22}^{(R)} & \ldots & a_{2n}^{(R)} \\ \hdashline \vdots & \vdots & \ddots & \vdots \end{array}\right]$$

Let $a_{ij}^{(r)}$ be the r-th row entry in the j-th column of the i-th group and let $\boldsymbol{a}_{ij} = [a_{ij}^{(1)} a_{ij}^{(2)} \ldots a_{ij}^{(R)}]^T$. First, for each (i,j) not in $\mathcal{G}_{n,\mu}$, we set $\boldsymbol{a}_{ij} = 0^R$. Next, we randomly chose $3n$ distinct values from the set $\mathcal{C} \triangleq \left\{[c_1,c_2,\ldots,c_R]^T \in (\mathbb{Z}_M)^R : \text{gcd}(c_1,c_2,\ldots,c_R) = 1\right\}$ and use these to set the

[3]Reference [1] proposes a design of matrix $\mathbf{A}_{\mu\times n}$ such that given $\boldsymbol{y} = \mathbf{A}\boldsymbol{d}$ for a k-sparse vector $\boldsymbol{d} \in \mathbb{R}^n$, a reconstruction $\hat{\boldsymbol{d}}$ can be obtained in $\mathcal{O}(k)$ steps by using a measurement vector of length $\mu = \mathcal{O}(k)$. A key requirement of this design is that both the locations of non-zero entries of $\mathbf{A}$ as well as their values may be arbitrarily chosen. In particular, the non-zero entries of $\mathbf{A}$ are chosen to be unit norm complex numbers.

values of $\boldsymbol{a}_{ij}$ for each edge (i, j) in $\mathcal{G}_{n,\mu}$. The assumption that $M^R/\zeta(R) \geq 3n$ ensures that such a sampling is possible. We skip the proof of Lemma 1 here and refer the reader to [1] for the proof.

Lemma 1. *[1, Lemma 6] For M large enough, $M^R/\zeta(R) \leq |\mathcal{C}| \leq M^R$.*

The output of the measurement is a $R\mu$-length vector $\boldsymbol{y} = \mathbf{A}\boldsymbol{d}$. Again, we partition $\boldsymbol{y}$ into μ groups of R consecutive rows each, and denote the i-th sub-vector as $\boldsymbol{y}_i$. Note each $\boldsymbol{y}_i \in \mathbb{R}^R$ follows the relation $\boldsymbol{y}_i = [a_{i1}a_{i2}\dots a_{in}]\boldsymbol{d}$.

Reconstruction algorithm: The decoding process is essentially equivalent to the "peeling process" to find 2-core in uniform hypergraph [22], [23]. The decoding takes place over $\mathcal{O}(k)$ iterations. The decoding algorithm is very similar to [1, Section IV-B]. In each iteration, we find one non-zero undecoded d_j with a constant probability after locating a right node that is connected to exactly one non-zero left node. After decoding the non-zero d_j for the current iteration, we cancel out the contribution of d_j from all measurements and proceed iteratively. To describe the peeling process, we define leaf nodes as follows.

Definition 2 (S-leaf node)**.** *A right node i is a leaf node for S if i is connected to exactly one $j \in S$ in the graph $\mathcal{G}_{n,\mu}$.*

First, the algorithm initializes the *reconstruction vector* $\hat{\boldsymbol{d}}(1)$ to the all zeros vector 0^n, the *residual measurement vector* $\tilde{\boldsymbol{y}}(1)$ to $\boldsymbol{y}$, and the *neighbourly set* $\mathcal{D}(1)$ to be the set of all right nodes for which y_i does not equal 0^R. In each iteration t, the decoder picks a right node $i(t)$ from the current neighbourly set $\mathcal{D}(t)$ and checks if only one left node contributes to the value of $(\tilde{y}(t))_{i(t)}$. If so, it identifies $i(t)$ as a *leaf* node, decodes delay value at the corresponding parent node, and updates $\mathcal{D}(t+1), \tilde{\boldsymbol{y}}(t+1)$, and $\hat{\boldsymbol{d}}(t+1)$ for the next iteration. The decoder terminates when the residual measurement vector becomes zero. See [1, Section IV-B] for a detailed description.

Next, we prove the performance guarantees of SHO-FA-INT as claimed in Theorem 1. Let $k = k(n)$ grow as a function of n. We show that the algorithm presented above correctly

reconstructs the vector $\hat{\boldsymbol{d}}$ with a high probability over the ensemble of matrices $\{\mathbf{A}_{R\mu\times n}\}$. To this end, we first note that if $\mu = \Omega(k)$, the ensemble of graphs $\{\mathcal{G}_{n,\mu}\}$ satisfies the following "many leaf nodes" as shown in the following lemma.

Lemma 2 (Many leaf nodes)**.** *[1, Lemma 2] Let S be a subset of the left nodes of the $\mathcal{G}_{n,\mu}$ and let $N(S')$ be the set of right neighbours of a set S'. If $|S| \leq k$ then with probability $1-\mathcal{O}(1/k)$, for every $S' \subseteq S$, $N(S')$ contains at least $|N(S')|/2$ S'-leaf nodes.*

Proof of Theorem 1: Let $S(\boldsymbol{d}) = \{j \in \{1, 2, \ldots, n\} : d_j \neq 0\}$. By Lemma 2, with probability $(1 - \mathcal{O}(1/k))$, all its subsets S' of $S(\boldsymbol{d})$ have at least twice as many leaf neighbours as the the number of elements in the S'. Therefore, in each iteration, the probability of picking a leaf node is at least half. Next, we note that in each iteration that we pick a leaf node, the probability of identifying as one and finding it's left neighbour correctly is 1. This is true because the weight vectors $\boldsymbol{a}_{ij}$'s corresponding different neighbours of a given right node i are different and for a leaf node i with the sole non-zero neighbour j, the output value $\boldsymbol{y}_i$ exactly equals to $d_j\boldsymbol{a}_{ij}$.

Next, we argue that if i is not a leaf node, then the probability of it being declared a leaf node in any iteration is $\mathcal{O}(1/n)$. Note that for this error event to occur for a right node i, it must be true that $\sum_{j'\in N(i)} d_{e'}a_{ij'} = d''a_{ij''}$ for some $d'' \in \mathbb{R}$ and j'' connected to i. Since all the measurement weights are chosen randomly, by Schwartz-Zippel lemma [24], [25], the probability of this event is $\mathcal{O}(1/n)$, which is $o(1/k)$.

Since the probability of picking a leaf node at any iteration is at least $1/2$, the expected number of iterations before a new leaf node is picked is upper bounded by 2. Since there are at most k non-zero d_j's, in expectation, the algorithm terminates in $\mathcal{O}(k)$ steps. Further, since the probability of finding a leaf in each iteration is independent of other iterations, by applying standard concentration arguments, the total number of iterations required is upper bounded by $2k$ in probability.

Finally, to compute the decoding complexity, note that each iteration requires a constant number arithmetic operations over vectors in $[M]^R$, which in turn can be decomposed into

$\mathcal{O}(R)$ arithmetic operations over integers. Therefore, the total number of integer operations required is $\mathcal{O}(Rk) = \mathcal{O}(k\lceil \log n/\log M\rceil)$. Finally, we note that since each left node in $\mathcal{G}_{n,\mu}$ has exactly 3 right neighbours which are picked uniformly among all right nodes and independently across different left nodes, with a high probability, each right node has no more than $4n/\mu$ left neighbours. This can be proved by first computing the expected number of left neighbours for a right node and then applying Chernoff bound to concentrate it to close to its expectation. This shows that, with a high probability, the number of non-zero entries in each row of $\mathbf{A}$ is $\mathcal{O}(n/k)$.

■

VII. The FRANTIC algorithm

A. Link Delay Estimation: We define a *path* $\boldsymbol{P}$ of length T over the network $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ as a sequence $(e_1, e_2, \ldots, e_T) = ((v_1, v_2), (v_2, v_3), \ldots (v_T, v_{T+1}))$ such that $e_t \in \mathcal{E}$ for $t = 1, 2, \ldots, T$. For a given path $\boldsymbol{P}$, we define the multiplicity $W(\boldsymbol{P}, e)$ of a link $e \in \mathcal{E}$ as the number of times $\boldsymbol{P}$ visits e. Let $\Delta(\boldsymbol{P})$ be the end-to-end delay for a path $\boldsymbol{P}$.

Definition 3 ($\boldsymbol{w}$-spanning measurement**).** *Given a measurement weight vector* $\boldsymbol{w} = [w_1 w_2 \ldots w_{|\mathcal{E}|}]$*, and a* $\boldsymbol{w}$-spanning measurement *is a path* $\boldsymbol{P} = (e_1, e_2., \ldots e_T)$ *in the network* $\mathcal{N}$ *such that* $\boldsymbol{P}$ *visits each* e *in* $\{e : w_e \neq 0\}$ *at least once.*

Definition 4 ($(\boldsymbol{w}, \boldsymbol{P})$-weighted measurement**).** *Given a measurement weight vector* $\boldsymbol{w}$ *and a* $\boldsymbol{w}$*-spanning measurement* $\boldsymbol{P} = (e_1, e_2, \ldots e_T)$*, a* $(\boldsymbol{w}, \boldsymbol{P})$-weighted measurement *is a path* $\boldsymbol{Q} = (e'_1, e'_2, \ldots e'_H)$ *in the network* $\mathcal{N}$ *such that* $W(\boldsymbol{Q}, e) = W(\boldsymbol{P}, e) + 2w_e$ *for each link* e*.*

Observe that the end-to-end delay for a $\boldsymbol{w}$-spanning measurement $\boldsymbol{P}$ is equal to $\Delta(\boldsymbol{P}) = \sum_{e\in\mathcal{E}} W(\boldsymbol{P}, e)d_e$, and that for a $(\boldsymbol{w}, \boldsymbol{P})$-weighted measurement is equal to

$$\Delta(\boldsymbol{Q}) = \sum_{e\in\mathcal{E}} W(\boldsymbol{Q}, e)d_e = \Delta(\boldsymbol{P}) + 2\sum_{e\in\mathcal{E}} w_e d_e. \tag{1}$$

Proof of Theorem 2: To prove Theorem 2, we start with a measurement matrix $\mathbf{A}$ drawn according to the SHO-FA-INT construction for Theorem 1. For each row of the measurement matrix, we construct two paths in the network - a spanning measurement and a weighted measurement. Next,

we subtract the end-to-end delay for the spanning measurement from the weighted measurement to get an output that is exactly twice the measurement output corresponding to the compressive sensing measurement using measurement matrix $\mathbf{A}$. Thus, we can apply the SHO-FA-INT reconstruction algorithm from Section VI to reconstruct the delay vector $\boldsymbol{d}_{\mathcal{E}}$. More precisely, Let $\mathbf{A}$ be a $R\mu \times n$ matrix drawn from the ensemble of Section VI, where $R = \mathcal{O}(\lceil \log n / \log M \rceil)$ and $n = |\mathcal{E}|$.

Measurement Design: Let $\boldsymbol{a}(i) = [a_{i1} a_{i2} \dots a_{in}]$ be the i-th row of $\mathbf{A}$. Consider network measurements $\boldsymbol{P}(i)$ and $\boldsymbol{Q}(i)$ defined as follows. Let $\boldsymbol{P}(i)$ be an $\boldsymbol{a}(i)$-*spanning measurement* obtained by picking the links in $\{e : \boldsymbol{a}(i) \neq 0\}$ one-by-one and finding a path from one link to another. By the definition of the diameter of the graph, there exists a path of length at most D between any pair of links. Therefore, there exists a path $\boldsymbol{P}(i) = ((v_1, v_2), (v_2, v_3), \dots, (v_T, v_{T+1}))$ of length $T = \mathcal{O}(Dn/k)$ that covers all the $\mathcal{O}(n/k)$ vertices that have non-zero components in $\boldsymbol{a}(i)$.

Next, let $\boldsymbol{Q}(i) = (e'_1, e'_2 \dots, e'_{T'})$ be a $(\boldsymbol{P}(i), \boldsymbol{a}(i))$-*weighted measurement* of length $T' = T + 2\sum_{e \in \mathcal{E}} a_e(i)$ as follows. Let $e'_1 = (v_1, v_2)$. If $a_{(v_1,v_2)}(i) \neq 0$, we traverse the edge (v_1, v_2) an additional $2a_{(v_1,v_2)}(i)$ times by going in the forward direction, *i.e.* on (v_1, v_2), and the reverse direction, *i.e.* on (v_2, v_1), an additional $a_{(v_1,v_2)}(i)$ times each. Thus, for $\tau = 1, 3, 5, \dots, 2a_{(v_1,v_2)}(i) + 1$, we set $e'_\tau = (v_1, v_2)$ and for $\tau = 2, 4, \dots, 2a_{(v_1,v_2)}(i)$, we set $e'_\tau = (v_2, v_1)$. Next, if $v_3 = v_1$, *i.e.*, we have already visited e_2, we traverse the link we traverse the link e_2 once more, else we traverse it $a_{(v_2,v_3)}(i) + 1$ times in the forward direction and $a_{(v_2,v_3)}(i)$ times in the reverse direction, *i.e.*, for $\tau = 2a_{(v_1,v_2)}(i) + 2, 2a_{(v_1,v_2)}(i) + 4, \dots, 2a_{(v_1,v_2)}(i) + 2a_{(v_2,v_3)}(i) + 2$, we set $e'_\tau = (v_2, v_3)$ and for $\tau = 2a_{(v_1,v_2)}(i) + 3, 2a_{(v_1,v_2)}(i) + 5, \dots, 2a_{(v_1,v_2)}(i) + 2a_{(v_2,v_3)}(i) + 1$, we set $e'_\tau = (v_3, v_2)$. We continue this process for each link (v_t, v_{t+1}) in the path $\boldsymbol{P}(i)$, , *i.e.*, if (v_t, v_{t+1}) has been visited already in either the forward or reverse direction by $\boldsymbol{Q}(i)$, we add it to $\boldsymbol{P}(i)$ only once, else, we traverse it an additional $a_{(v_t,v_{t+1})}(i)$ times in each direction. Therefore, $\boldsymbol{Q}(i)$ visits every edge $e \in \mathcal{E}$ a total of $2a_e(i)$ times more than $\boldsymbol{P}(i)$ does.

Reconstructing $\boldsymbol{d}_{\mathcal{E}}$: Next, we measure the end-to-end delays for the paths $\boldsymbol{P}(i)$ and $\boldsymbol{Q}(i)$ for each $i = 1, 2, \dots, R\mu$ and let $y_i = (\Delta(\boldsymbol{Q}(i)) - \Delta(\boldsymbol{P}(i)))/2$. From equation (1), it follows that $y_i = \sum_{e \in \mathcal{E}} a_{ie} d_e$. Note that this exactly equals the output of a compressive sensing measurement

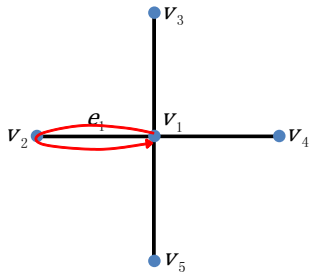


Figure 8. Isolated Node: For a subgraph with 5 vertices and 4 links, all vertices are congested. v_1 is not isolated since all of its neighbors are congested. Suppose there is a local loop involving v_1, v_2, and e_1. For link measurement, only the delay of e_1 is added to the weighted measurement. However, for the node measurement, the delays of v_1 and v_2 are both added to the weighted measurement. The delay of v_2 will not be canceled by the corresponding spanning measurement.

with $\boldsymbol{d}$ as the input vector, $\mathbf{A}$ as the measurement matrix, and $\boldsymbol{y}$ and the measurement output vector. Using this observation, we input the vector $\boldsymbol{y}$ to the SHO-FA-INT algorithm to correctly reconstruct $\boldsymbol{d}$ with probability $1 - \mathcal{O}(1/k)$. The guarantees on the decoding complexity follow from the decoding complexity of the SHO-FA-INT algorithm and that on the total number of hops follows by noting that each link in a measurement path may be visited at most $2M$ times. ■

B. Node Delay Estimation: The measurement design and the decoding algorithm for node delay estimation proceeds in a similar way to the link delay estimation algorithm of Section VII. The difference here is that instead of assigning weights to links in a path, our design assigns weights to nodes in a path by visiting each node repeatedly. We skip the proof of Theorem 3 here as it essentially follows from the technique used in the proof of Theorem 2. The only difference is that for node delay estimation we add the isolation assumption. If there exists one congested node, $v \in \mathcal{V}$, whose neighbors are all congested nodes, then we are not able to generate the measurement involving v by subtracting the weighted measurement from the spanning measurement. The reason is that each local loop involving v adds one more delay corresponding to one of its congested neighbor. However, this problem doesn't happen in the edge delay measurements. (See Fig. 8)

VIII. Exploiting network structure

A. Reducing Path Lengths through Steiner Trees:

One drawback of the approaches presented in the previous section is that even though on an average, each row of $\mathbf{A}$ contains only $\mathcal{O}(n/\rho k)$ non-zero entries, our upper bound on the path length relies on worst case pairwise paths for each pair of successive edges to be measured. In

this Section, we propose a Steiner Tree based approach to design the measurement paths given a measurement matrix $\mathbf{A}$.

Definition 5 (Steiner Tree)**.** *Let $\mathcal{S} \subseteq \mathcal{V}$. We say that $\mathcal{T} \subseteq \mathcal{E}$ is a Steiner Tree for $\mathcal{S}$ if $\mathcal{T}$ has the least number of edges among all subsets of $\mathcal{E}$ that form a connected graph that is incident on every $v \in \mathcal{S}$. Let $L(\mathcal{S})$ be the length of a Steiner Tree for $\mathcal{S}$.*

For every $s \in \mathbb{Z}^+$, let

$$L^*(s) \triangleq \max_{\substack{\mathcal{S}:\mathcal{S}\subseteq\mathcal{V} \\ |\mathcal{S}|\leq s}} L(\mathcal{S}).$$

Note that, in general, $L^*(s) \leq Ds$. Further, in many graphs of practical interest, $L^*(s) \ll Ds$. For example, in a line graph with n vertices, $L^*(s)$ is at most n, while Ds maybe as large as $\mathcal{O}(ns)$. Using this observation, we may further improve the performance guarantee of our algorithm. We note that it suffices to find a Steiner Tree that passes through all links specified by a given row of the measurement matrix $\mathbf{A}$. Also, we already know that, with a high probability, the number of non-zero entries in each row of $\mathbf{A}$ is $\mathcal{O}(n/\rho k)$. Thus, in general, the number of links traversed by each link (or node) delay measurement is $\mathcal{O}(L^*(s))$ where $s = \mathcal{O}(|\mathcal{E}|/\rho k)$ (or $\mathcal{O}(|\mathcal{V}|/\rho k)$ respectively) is the number of non-zero entries in the measurement. This proves the following assertion.

Theorem 4 (Network tomography for link/node congestion using Steiner Trees)**.** *For the setting of Theorem 2, the number of links of $\mathcal{N}$ traversed by each measurement of* FRANTIC *is at most $\mathcal{O}(L^*(s))$ where $s = \mathcal{O}(|\mathcal{E}|/\rho k)$ is the number of non-zero entries in the measurement and the total number of hops for each measurement is $\mathcal{O}(ML^*(s))$.*

Remark: There exist polynomial-time approximation schemes with a performance ratio decreased from 2 to 1.55 by a series of works [26]–[33].

B. Average length of Steiner Trees:

In Theorem 4, we analyzed the length of measurement paths in terms of the worst-case length of Steiner trees that contain an arbitrary subset of s links (resp. nodes). However, on an average,

however, this may be too conservative an estimate.

Definition 6 (Average length of Steiner tree)**.** *For every* $s \in \mathbb{N}^+$*, let*

$$\overline{L}(s) \triangleq \frac{\sum_{\mathcal{S}:\mathcal{S}\subseteq\mathcal{V}} \atop {|\mathcal{S}|=s} L(\mathcal{S})}{|\{\mathcal{S} \subseteq \mathcal{V} : |\mathcal{S}| = s\}|}$$

denote the average length of Steiner tree.

In the example shown in Fig. 9, we argue that, with a high probability, the length of paths required is upper bounded by $\overline{L}(s)$ which may be significantly smaller than $L^*(s)$.

C. Network decomposition:

Since we already know the topology of the network, exploring the structure of the topology may help us to reduce the path length of each measurement. In Fig. 10, we illustrate how to reduce the length of Steiner tree by network decomposition.

IX. Acknowledgements

The work described in this paper was partially supported by a grant from University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-02/08),

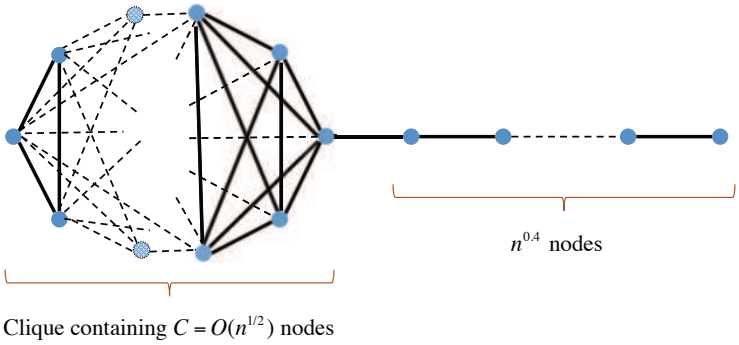


Figure 9. Worst-case vs Average length of Steiner trees: Consider a network of n links. The network has two parts - a clique consisting of $C = (1 + \sqrt{1 + 8(n - n^{0.4})})/2$ fully connected nodes and a line consisting of $n^{0.4}$ nodes. Let the number of congested links in the network be $k = n^{0.95}$. Thus, each measurement path has to cover a set of links of size $\mathcal{O}(n^{0.05})$ specified by SHO-FA-INT . In the worst case, such a set can include the two ends of the linear part. Thus, in the worst case, the length of the Steiner tree can exceed $n^{0.4}$. However, we note that the SHO-FA-INT algorithm picks the measurement nodes uniformly at random. Thus, the probability of picking even one edge from the linear part of the network is $\mathcal{O}(n^{0.45}/n^{0}.5) = \mathcal{O}(n^{-0.5})$ by the union bound. Therefore, on an average, the length of Steiner tree is at most $\mathcal{O}(n^{-0.5} \times n^{0.4}) = \mathcal{O}(n^{0.35})$, which is lower than the worst-case length of a Steiner tree covering $\mathcal{O}(n^{0.05})$ links in the network.

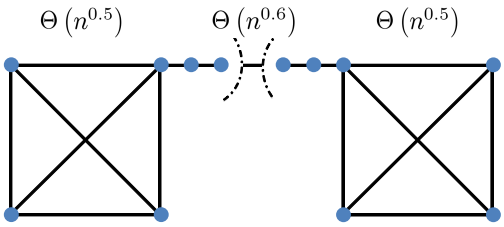


Figure 10. Network Decomposition: The network consists of three parts – two complete graphs with $\Theta\left(n^{0.5}\right)$ vertices and a line graph with $\Theta\left(n^{0.6}\right)$ vertices. It follows that there are $\Theta(n)$ links in each of the two complete graphs and $\Theta\left(n^{0.6}\right)$ links in the line graph. For each link measurement, with high probability, two links involved locate in each of two complete graphs. Therefore, the average length of Steiner tree is at least $\Theta\left(n^{0.6}\right)$. If we decompose the original network into two subgraphs as shown in the figure and do the link delay estimation on them separately, the average length of Steiner tree becomes at most $\Theta\left(n^{0.5}\right)$ which is smaller than $\Theta\left(n^{0.6}\right)$.

a grant from the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies, and an SHIAE grant.

REFERENCES

[1] M. Bakshi, S. Jaggi, S. Cai, and M. Chen, "SHO-FA: Robust compressive sensing with order-optimal complexity, measurements, and bits," *ArXiv.org eprint archive*, 2012, arXiv:1207.2335v1 [cs.IT].

[2] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of the American Statistical Association*, pp. 365–377, 1996.

[3] T. Bu, N. Duffield, F. Presti, and D. Towsley, "Network tomography on general topologies," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 30, no. 1, 2002, pp. 21–30.

[4] Y. Chen, D. Bindel, H. Song, and R. Katz, "Algebra-based scalable overlay network monitoring: algorithms, evaluation, and applications," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 5, pp. 1084–1097, 2007.

[5] J. Kleinberg, "Detecting a network failure," in *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, 2000, pp. 231–239.

[6] H. Nguyen and P. Thiran, "Using end-to-end data to infer lossy links in sensor networks," in *Proc. IEEE Infocom*, 2006, pp. 1–12.

[7] Y. Zhao, Y. Chen, and D. Bindel, "Towards unbiased end-to-end network diagnosis," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 6, pp. 1724–1737, 2009.

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

[9] W. Xu, E. Mallada, and A. Tang, "Compressive sensing over graphs," in *INFOCOM, 2011 Proceedings IEEE*, april 2011, pp. 2087 –2095.

[10] M. Firooz and S. Roy, "Network tomography via compressed sensing," in *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, 2010, pp. 1–5.

[11] M. Wang, W. Xu, E. Mallada, and A. Tang, "Sparse recovery with graph constraints: Fundamental limits and measurement construction," in *INFOCOM, 2012 Proceedings IEEE*, march 2012, pp. 1871 –1879.

[12] M. Cheraghchi, A. Karbasi, S. Mohajer, and V. Saligrama, "Graph-constrained group testing," *Information Theory, IEEE Transactions on*, vol. 58, no. 1, pp. 248 –262, jan. 2012.

[13] N. Harvey, M. Patrascu, Y. Wen, S. Yekhanin, and V. Chan, "Non-adaptive fault diagnosis for all-optical networks via combinatorial group testing on graphs," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, may 2007, pp. 697 –705.

[14] Y. Xuan, Y. Shen, N. Nguyen, and M. Thai, "Efficient multi-link failure localization schemes in all-optical networks," *Communications, IEEE Transactions on*, vol. PP, no. 99, pp. 1 –8, 2013.

[15] S. Ahuja, S. Ramasubramanian, and M. Krunz, "Srlg failure localization in optical networks," *Networking, IEEE/ACM Transactions on*, vol. 19, no. 4, pp. 989 –999, aug. 2011.

[16] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Kluwer Academic Publishers, 1996, pp. 153–181.

[17] E. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406 –5425, dec. 2006.

[18] E. J. Candès, J. K. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information." *IEEE Transactions on Information Theory*, pp. 489–509, 2006.

[19] D. L. Donoho, "Compressed sensing." *IEEE Transactions on Information Theory*, pp. 1289–1306, 2006.

[20] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, sept. 2008, pp. 798 –805.

[21] W. Xu and B. Hassibi, "Efficient compressive sensing with deterministic guarantees using expander graphs," in *Information Theory Workshop, 2007. ITW '07. IEEE*, sept. 2007, pp. 414 –419.

[22] M. T. Goodrich and M. Mitzenmacher, "Invertible bloom lookup tables," *ArXiv.org e-Print archive, arXiv:1101.2245 [cs.DB]*, 2011.

[23] M. Molloy, "Cores in random hypergraphs and boolean formulas," *Random Struct. Algorithms*, vol. 27, no. 1, pp. 124–135, Aug. 2005.

[24] J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM*, vol. 27, no. 4, pp. 701–717, 1980.

[25] R. Zippel, "Probabilistic algorithms for sparse polynomials," *Proceedings of the International Symposiumon on Symbolic and Algebraic Computation*, pp. 216–226, 1979.

[26] H. Takahashi and A. Matsuyama, "An Approximate Solution for the Steiner Problem in Graphs," *Math. Jap.*, vol. 24, pp. 537–577, 1980.

[27] A. Zelikovsky, "An 11/6-Approximation Algorithm for the Network Steiner Problem," *Algorithmica*, vol. 9, pp. 463–470, 1993.

[28] P. Berman and V. Ramaiyer, "Improved Approximations for the Steiner Tree Problem," *J. of Algorithms*, vol. 17, pp. 381–408, 1994.

[29] A. Zelikovsky, “Better Approximation Bounds for the Network and Euclidean Steiner Tree Problem,” *Technical report CS-96-06, University of Virginia*, 1993.

[30] H. J. Prömmel and A. Steger, “Rnc-approximation algorithms for the steiner problem,” in *Proc. STACS’97*, 1997, pp. 559–570.

[31] M. Karpinski and A. Zelikovsky, “New Approximation Algorithms for the Steiner Tree Problem,” *Journal of Combinatorial Optimizaion*, vol. 1, pp. 47–65, 1997.

[32] S. Hougardy and H. J. Prömel, “A 1.598 approximation algorithm for the steiner problem in graphs,” in *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, 1999, pp. 448–453.

[33] G. Robins and A. Zelikovsky, “Improved steiner tree approximation in graphs,” in *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, 2000, pp. 770–779.