Sparsity without the Complexity: Loss Localisation Using Tree Measurements

Vijay Arya, Darryl Veitch

Sparsity without the Complexity: Loss Localisation using Tree Measurements

Vijay Arya[1] and Darryl Veitch[2]

[1] IBM–Research, D4 Mayanta, Bangalore 560045, India, vijay.arya@in.ibm.com
[2] Dept. of EEE, The University of Melbourne, Australia, dveitch@unimelb.edu.au

Abstract. We study network loss tomography based on observing average loss rates over a set of paths forming a tree – a severely underdetermined linear problem for the unknown link loss probabilities. We examine in detail the role of sparsity as a regularising principle, pointing out that the problem is technically distinct from others in the compressed sensing literature. While sparsity has been applied in the context of tomography, key questions regarding uniqueness and recovery remain unanswered. Our work exploits the tree structure of path measurements to derive sufficient conditions for sparse solutions to be unique and the condition that ℓ_1 minimization recovers the true underlying solution. We present a fast single-pass linear algorithm for ℓ_1 minimization and prove that a minimum ℓ_1 solution is both unique and sparsest for tree topologies. By considering the placement of lossy links within trees, we show that sparse solutions remain unique more often than is commonly supposed. We prove similar results for a noisy version of the problem.

Keywords: network monitoring, network tomography, loss inference, tree topology, sparsity, ℓ_1 regularization, compressed sensing.

1 Introduction

Network operators and end applications alike would like to localize abnormally lossy links or *loss hotspots*, but how can this be achieved when internal access to the network is limited? Consider a set of nodes instrumented as active probing sources or receivers, generating flows of probes over a set of paths in the network to measure loss. The intersections of these paths result in a set of relations for mutual consistency of the measured path loss probabilities in terms of the constituent link loss probabilities that one would like to recover. This is a network tomography problem, defined over the measurement sub-network traversed by the probes, which can be expressed as a linear system. This system is in general severely under-determined: instead of a unique solution for the link loss rates, an entire family of solutions is consistent with the observed path measurements.

One way to select a particular solution from the family, that is one regularising principle, is *sparsity*: preferring the solution with the smallest number of lossy links. Sparsity is in keeping with Occam's razor which seeks the simplest explanation to a given set of observations, and is a natural fit to the assumption

Sparsity K	**Classical CS Results**	**Tree Loss Tomography Results**
Signal x	$\mathbb{R}^n$	link loss vector $\mathbb{R}^{+n}$
m measurements	$m \ll n$, variable	$m = n(1 - 1/c) + o(1)$, **fixed**
Matrix A	$m \times n$, $a_{ij} \sim \mathcal{N}(0, 1/m)$	$m \times n$ binary matrix representing the tree measurement topology
Uniqueness	every $2K$ columns of A independent	Every branch node has at least 2 lossless incident links ($K \leq m$)
Efficient Recovery	$m = O(K \log(n/k))$, RIP conditions on A	Every branch node has at least 1 lossless child link ($K \leq m$)

Table 1. A comparison of problem statements and results for a typical CS problem compared to loss tomography over a tree (with c children under each node).

that hotspots are rare. It also sits well with an operational need to provide a short list of potential hotspots worthy of closer attention. This paper examines in detail the role of sparsity in network loss tomography in the important special case of a tree-like measurement topology or *measurement matrix* A. Trees have been considered in a few prior works treating sparsity in loss tomography (see section 2), but mostly in an implicit sense, as a default special case of general networks. In this paper we show that the structure of trees can be exploited to allow an essentially complete picture to be obtained including conditions for uniqueness of sparse solutions, the relationship of sparse to ℓ_1 solutions, and the condition that minimum ℓ_1 recovers the true underlying solution.

Tree measurement topologies arise in several practical contexts – for instance unicast path measurements between a web server and its clients form a tree. Thus information on locations of hotspots could be used to direct clients to replica servers or have the hotspots resolved in cooperation with the concerned ISP [1]. Moreover, since any general measurement infrastructure can be configured for use as a tree, our results are relevant in practice. Furthermore, loss rates inferred from multiple intersecting trees over the same infrastructure can also be used to quickly obtain important partial information about general measurement topologies/matrices. Exploiting tree solutions and insights to gain purchase on the more general problem is a new direction and the subject of ongoing work. An early result in this direction is Lemma 1 in section 4.3.

There are three main differences between our problem setting and that of compressed sensing (CS), which has in recent years exploited sparsity for signal recovery (see Table 1). First, network tomography deals with positive quantities such as link loss probabilities and delays, whereas CS generally treats real valued signals. Second, in CS one inquires after the nature of the measurement matrix A, and its size (number of observations m) needed to recover a solution of given sparsity K uniquely. In network tomography both the nature and size of A are highly constrained by the availability of measurement nodes, and the lack of control over the packet routing between them. In the case of a tree, adding a new 'measurement' is non-trivial as it implies installing an active probe receiver in a new location. Finally, the measurement matrices studied in CS are designed

to satisfy strong technical conditions such as the restricted isometry property (RIP [2, 3]). In contrast, for a tree the measurement matrix A is given rather than designed, and has a specific dependence structure. Studying sparse solutions over trees is quite different from 'traditional CS' approaches, however, the key questions of interest are the same: **hardness, uniqueness, and recovery**.

Compared to prior CS work, our task is nontrivial in that we do not benefit from properties such as RIP and must develop fresh techniques, but easier in that the structure of trees is simple and powerful. As we see below, the net result is that the tree context is more tractable, enabling detailed and complete solutions with desirable properties. Our main contributions include:

Hardness (Complexity of computing a sparse solution): Recovering the sparsest or min ℓ_0 (pseudo) norm solution of under-determined systems is in general NP-hard [4]. For a tree we show that the sparsest solution(s) may be characterised precisely and found with a fast linear time algorithm.

Uniqueness (Conditions for sparsest solution to be unique): Provided the number of lossy links at every internal/branch node with node degree g is at most $g-2$, the sparsest solution is unique. This result takes into account the locations of loss within the tree, and shows that uniqueness may hold for much higher values of sparsity K than the worst case analysis typically used in CS would suggest. When solutions of a given sparsity are not unique, the alternative solutions can be precisely localised and characterised.

Recovery (Conditions that min ℓ_1 solution is the true solution): For the general problem, the min ℓ_1 norm solution does not always have the min ℓ_0 norm and need not even be unique. For the tree problem, we show that min ℓ_1 solution always has the min ℓ_0 norm and is unique. Provided every internal/branch node has at least one lossless child link, the min ℓ_1 recovers the true underlying solution. We define the 'UpSparse' algorithm, a fast single-pass linear-time algorithm which outputs the min ℓ_1 solution. For the general problem, the min ℓ_1 solution is recovered through a linear program (cubic complexity).

Since in practice only a finite number of probes can be sent, the measured path loss probabilities can only be known approximately. We formulate and study a 'noisy' version of the problem that addresses this key practical concern. As before, we exploit the nature of the tree, and characterise the minimal norm solutions and present fast algorithms to recover them. The noisy problem is briefly discussed in section 4.4 with results in the technical report [5].

We begin by discussing related work in section 2. Section 3 presents the general solution for the hotspot localisation problem in trees. Section 4 characterises the sparse and min ℓ_1 solutions and shows how to find them. Section 5 compares our algorithms with CS optimization techniques. Section 6 presents experimental results where we explore the relationship between the sparse and true solutions.

2 Related Work

Several problems [1, 6–11], all related to inferring link parameters using either unicast or multicast path measurements, have been studied under the purview

of Network Tomography. Whereas multicast measurements utilise observations at the per-probe level, the unicast tomography problem works with average observations of paths and reduces to an under-determined linear system in terms of unknown link parameters. However the most common approach to solve an underdetermined system, namely choosing the min ℓ_2 norm solution, may not be suitable when the link quantity is concentrated at particular locations. For example, for loss inference, it tends to spread the loss over all links in the network.

More recent work borrows techniques of recovering sparse solutions to underdetermined linear systems from compressed sensing (CS) [2,3,12]. In CS, a fixed but randomly generated (generally Gaussian) matrix A is used to 'measure' an unknown signal $\boldsymbol{x}$ as $\boldsymbol{y}_{m\times 1} = A_{m\times n}\boldsymbol{x}_{n\times 1}$; $m \ll n$ so that x is underdetermined. CS results show that a minimally sparse $\boldsymbol{x}$ can be recovered with high probability (i.e. for most A) using ℓ_1 minimization [13]. As outlined above, network loss tomography is quite different: $\boldsymbol{y}$ represents the path observations, $\boldsymbol{x}$ the unknown link parameters, and A, which determines which paths traverse each link, cannot be chosen freely and has unknown properties in general.

Despite these differences, ℓ_1 minimization has been used as a black box to recover sparse solutions in tomography. In [9], Bayesian experimental design is used to determine the set of paths to measure in a network and a variant of ℓ_1 minimization is used to infer link parameters. In [10], variance in path measurements across multiple measurement intervals is used to identify a prior, and an ℓ_1 minimization formulation from [9] is used to find a sparse solution close to it. In [1], path measurements between a server and its clients are used to recover link loss rates by using sampling, Bayesian inference, and a variant of ℓ_1 minimization. None of these works provide insight into the nature of the sparse or ℓ_1 solutions, how they interact, or their uniqueness, the central focii of our work. In [6], locations of 'bad' network links is inferred from path measurements forming a tree. For this, each path is classified as 'good' (0) or 'bad' (1) and the smallest set of bad tree links consistent with the binary path observations is recovered. In section 6, we see that this two-step approach fails to recover the true locations of hotspots more often than our approach that directly recovers a minimally sparse link loss solution. In addition, we recover both the locations and loss rates of lossy links. Given a measurement matrix, [14] uses expander graphs [15] to determine conditions for recovering unique sparse solutions in networks. However, for trees expander graphs do not bring any additional insight.

The work most closely related to our own is [16], which answers some of the key questions for CS over graphs. The key difference is that we work with trees instead of general networks. The simpler tree topology enables far greater insight into the sparse and ℓ_1 solutions, and allows explicit solutions and fast algorithms to be defined. In [16] the authors determine the number of random measurements over underlying network paths needed to uniquely recover sparse link solutions. Random measurements however are difficult to justify in the tomography context. Conversely, for a given measurement matrix they provide upper bounds on the number of lossy links consistent with uniqueness of the sparsest solution. These bounds are quite restrictive for trees. For example for any ternary tree,

irrespective of its size, the largest allowed number of lossy links is 2, and for a binary tree the price of uniqueness guarantee is that only a single link may be lossy. In section 6, we see that for a ternary tree with 25 links, even when 4 links are lossy, the sparsest solution is still unique for 95% of feasible link loss vectors, and the proportion grows with tree size. In section 4.3, we also show how the recovery of a K-sparse vector relates to the degree of the measurement graph.

3 The Tree Hotspot Problem and Solution

In this section we describe how we model the loss process over a tree, and how to formulate the resulting problem as a linear system. We then solve the system formally, and make some preliminary observations.

Tree Model Let $\mathcal{T} = (V, L)$ denote the logical tree consisting of a set of nodes V and links L. Let $\mathcal{O} \in V$ denote the root node, $R \subset V$ be the set of leaf nodes, and $I = V \setminus \{\mathcal{O} \cup R\}$ the set of internal nodes. A link is an ordered pair $(j, k) \in \{V \times V\}$ representing a logical link (one or more physical links) from node j to node k. For each node except the root there is a unique node $j = f(k)$, the father of k, such that $(j, k) \in L$. The set of children of a node k is denoted by $c(k)$, thus $c(k) = \{j \in V : (j, k) \in L\}$. All nodes have at least two children, except the root (just one) and the leaves (none). The depth of a node is the number of links in the (unique) path of ancestors leading to the root. By level l of a tree we mean the set of nodes of depth l, with the root being of depth zero. We denote the *height* of the tree (the depth of the deepest leaves) by H. The *top link* is the unique link adjacent to the root node.

For convenience, we refer to link $(f(k), k)$ simply as link k, and similarly, we also use I to refer to the set of *internal links* corresponding to the internal nodes, R to refer to the leaf links as well as nodes, and so on. Let n denote the number of links in the tree, $m = |R|$ the number of leaves, and $d = n - m$ the number of internal links. From each leaf there is a unique path to the root, so m is also the number of paths. Clearly $n \geq m + 1$. It is convenient to label nodes/links as follows: First, the leaf nodes are labelled by $k = 1, 2 \ldots m$ from left to right. Then beginning with the child of the root the counting continues in a *preorder* traversal of the internal nodes of the tree (recursively: node, left subtree right subtree). With this convention, the labels of leaf nodes can double as convenient path labels. In other words, path j terminates at leaf node j, with paths labelled as $j = 1, 2 \ldots m$ from left to right. Examples are given in the figures.

The topology of tree is captured by the $m \times n$ measurement matrix A, where the entry $A_{jk} = 1$ if link k forms part of the path j, 0 otherwise. Row j of the matrix gives the links in path j, and column k gives the paths that cross link k.

Modelling Link Loss The marginal probability of loss on link k is given by $b_k \in [0, 1]$, and we denote the $(n \times 1)$ vector of loss probabilities over all links by $\mathbf{b}$. We assume stationarity so that $\mathbf{b}$ is constant. As in all prior work, spatial independence is assumed, i.e., all link loss processes are mutually independent. It follows that the path loss probability is easily expressed via the product of the link passage probabilities: $p_j = 1 - \prod_{k:A_{jk}=1}(1 - b_k)$ where the product is

over the links on path j. We assume that we have access, through measurements based on a large number of probes, to the exact path loss probability vector, $\mathbf{p} = [p_1, p_2, \ldots, p_m]^T$.

3.1 System Solution

Define the *addloss function* as $\mathcal{L}(b) = -\log(1-b),\ b \in [0,1)$. We write $x_k = \mathcal{L}(b_k)$ and $y_j = \mathcal{L}(p_j)$. Since $\mathcal{L}$ is a monotonically increasing function, mapping $[0,1)$ to $[0,\infty)$, the link loss vector $\mathbf{b}$ is replaced by the equivalent link *addloss vector* $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$, and similarly $\mathbf{p}$ is replaced by $\boldsymbol{y} = [y_1, y_2, \ldots, y_m]^T$. The relation $p_j = 1 - \prod_{k:A_{jk}=1}(1-b_k)$ is now $y_j = \sum_{k:A_{jk}=1} x_k$, and the relationship between path and link loss takes the linear form

$$\boldsymbol{y} = A\boldsymbol{x}, \quad x_k \geq 0, y_j \geq 0. \tag{1}$$

The term 'addloss' is justified by the additive nature of link addloss, together with the fact that values of x_k and y_j can still be interpreted directly as loss for many purposes. In particular 0 addloss implies zero loss. Since we use addloss exclusively in this paper, we will use 'loss' as a shorthand for addloss.

$$A = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right] = \left[\begin{array}{c|c} I_m & A_I \end{array}\right]$$

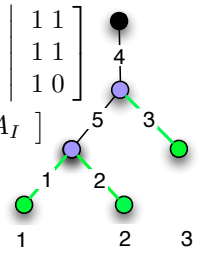

Fig. 1. A tree with 3 leaves (and paths) and 2 internal links, with its measurement matrix A. Receiver links correspond to the identity matrix I_m.

Consider the tree in figure 1 together with its measurement matrix A, where the vertical divider separates the m columns corresponding to receiver links $\boldsymbol{x}_R = [x_1, x_2, \ldots, x_m]^T$ on the left, from those of the internal links $\boldsymbol{x}_I = [x_{m+1}, \ldots, x_n]^T$ on the right. The $(m \times m)$ identity matrix I_m in the left appears because each leaf link belongs to just one path (and because of our link and path naming conventions). This is true in general for any tree, and we may partition any measurement matrix into I_m and the $(m \times d)$ matrix A_I that shows how internal links contribute to paths. We can now rewrite (1) as

$$\boldsymbol{y} = A\boldsymbol{x} = \begin{bmatrix} I_m & A_I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_R \\ \boldsymbol{x}_I \end{bmatrix} = \boldsymbol{x}_R + A_I\boldsymbol{x}_I \Rightarrow \boldsymbol{x}(\boldsymbol{x}_I; \boldsymbol{y}) = \begin{bmatrix} \boldsymbol{x}_R \\ \boldsymbol{x}_I \end{bmatrix} = \begin{bmatrix} \boldsymbol{y} - A_I\,\boldsymbol{x}_I \\ \boldsymbol{x}_I \end{bmatrix} \tag{2}$$

It is clear that A has full rank m, as its column rank is clearly at least m due to the embedded identity matrix, and row and column rank are equal. The general solution will therefore have $d = n - m$ free parameters. Since there are also d internal links, it is convenient to select $\boldsymbol{x}_I$ to span this space, in terms of which a formal solution can be immediately written as above where the path observation vector $\boldsymbol{y}$ appears as a parameter.

The choice of $\boldsymbol{x}_I$ as the independent variables has the advantage of making $\boldsymbol{y}$ appear in a simple way in the solution. For e.g., setting $\boldsymbol{x} = 0$, it is clear that $\boldsymbol{y} = \mathbf{0}$ is the corresponding observation. Setting $\boldsymbol{x}_I = 0$, we see that $\boldsymbol{y} = \boldsymbol{x}_R$, which also reveals that any set of (non-negative) observations is possible in general. We call $\boldsymbol{x}_I = 0, \boldsymbol{x}_R = \boldsymbol{y}$, the *receiver solution.*

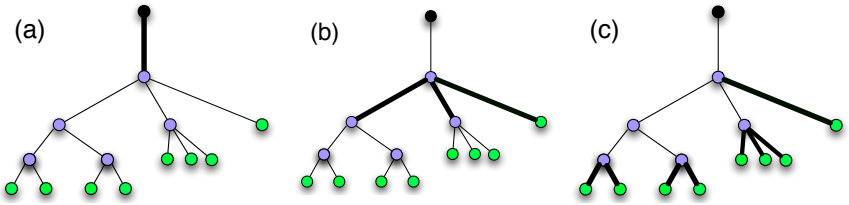

Fig. 2. Example of ambiguity in the locations of lossy links (bold), each of loss x. In each case, receiver vector $\boldsymbol{y} = [x, \ldots, x]^T$, but the no. of lossy links varies - 1, 3, and 8.

4 Regularizing the Solution using Sparsity

Equation (2) is a d-dimensional family of loss solutions all equally consistent with a single observed $\boldsymbol{y}$. From the point of view of an observer whose end goal is to identify a unique set of candidate loss hotspots, this represents a significant and problematic *ambiguity*. Our main regularising principle is that of *sparseness*, i.e. minimizing the number K of lossy links which are consistent with any observed $\boldsymbol{y}$. A smaller number is preferred because it is more likely under a priori assumption that loss is rare, and focusses attention on a smaller number of candidate problem links, which has practical advantages. If in fact solutions are sparse, i.e. given a bound on the sparsity K, we wish to determine the conditions under which a sparsest solution is the unique solution consistent with observed $\boldsymbol{y}$.

Consider figure 2 which gives three (of several) possible solutions consistent with the observations $\boldsymbol{y} = [x, \ldots, x]^T$, $x > 0$. Figure 2(a) shows the solution $\boldsymbol{x}_I = [x, 0, \ldots, 0], \boldsymbol{x}_R = 0$ where only the top link is lossy while figure 2(c) shows the receiver solution $\boldsymbol{x}_I = 0, \boldsymbol{x}_R = \boldsymbol{y}$. If sparsity $K \leq 1$, then 2(a) is the unique sparsest solution consistent with $\boldsymbol{y}$.

Finding the sparsest solution is equivalent to minimizing the ℓ_0 (pseudo) norm of $\boldsymbol{x}$: $K = ||\boldsymbol{x}||_0 = \sum_{i=1}^{n} |x_i|^0 = \sum_{i:x_i>0} 1$. Results from CS have shown that minimizing with respect to ℓ_1 norm often identifies solutions with $\min \ell_0$ norm but at a lower computational cost. We therefore also explore ℓ_1 below, both in its own right, and as a secondary principle which can be used to further reduce ambiguity. Whilst a priori information on the likely locations of lossy links within the topology may be available in some contexts, this is not always so. In this paper we treat the case where there is no such information, corresponding informally to a uniform prior over all links.

4.1 Local Regularisation

It is useful to understand ambiguity in a *local complex*. This is a 'building block' consisting of an internal/branch node and its adjacent links. We will explore it using the two level tree of figure 3(a), where link and path labels have been dropped in favor of their loss values. The general solution corresponding to the

observed $\boldsymbol{y}$ is $\boldsymbol{x}' = [y_1 - x, y_2 - x, \ldots, y_m - x, x]^T$, parameterised by $x \in [0, y_{\min}]$, where $y_{\min} = \min_j y_j$ is the smallest path loss. We examine sparsity in the local complex as a function of the parameter $x \in [0, y_{\min}]$.
$y_{\min} = 0$: the family collapses to a unique solution, $x = 0$.
$y_{\min} > 0$: x is not uniquely determined by $\boldsymbol{y}$ and we speak of an *ambiguous complex*. At $x = 0$ the internal link is lossless and the child link losses are maximized. We call this the *downstate*, and it has sparsity $K = m$. As x increases over the range $x \in (0, y_{\min})$ loss is 'pulled up' equally and in parallel from each child link to the internal link. In these *mixed states* all links are lossy and sparsity is $K = m + 1$, the largest possible. This upward transfer of loss ceases at the *upstate* when $x = y_{\min}$, where suddenly the loss on all $m_{\min}$ links sharing the minimum value $y_{\min}$ becomes zero, and K drops to $K = m - m_{\min} + 1$.

To summarise, requiring sparsity excludes the mixed states, singling out the downstate ($x = 0$) and upstate ($x = y_{\min}$). If $y_{\min}$ is not unique ($m_{\min} > 1$) then the upstate solution is the unique sparsest solution. For example when moving from figure 2(b) to figure 2(a), K drops locally around the top internal node from 3 to 1 as the complex moves from the downstate to the upstate. If instead $m_{\min} = 1$, then both the upstate and downstate have sparsity $K = m$, so ambiguity, though greatly reduced, remains.

If instead of sparsity we consider ℓ_1 the conclusions are similar but ambiguity vanishes. The ℓ_1 norm of the local complex illustrated in figure 3(a) is given by $||\boldsymbol{x}'||_1 = \sum_{k=1}^{n} |x_k| = \sum_{j=1}^{m} y_j - x(m-1)$. It is clearly minimized by setting x as large as possible, namely $x = y_{\min}$, i.e. the upstate. Hence for the ℓ_1 norm upstate is always both optimal and unique, whereas for ℓ_0 it is always optimal but not always unique. A simple example with $\boldsymbol{y} = [2, 3, 4]$ is shown in Figure 3(b).

4.2 Global Regularisation

The local ambiguity above can occur centered on any internal node in the tree. A natural question is whether global effects may resolve local ambiguities, or alternatively result in new forms of global ambiguities. Consider the scenario of figure 3(c) where the original sparsity is K. The lower complex is initially in a downstate (left) with $m_{\min} = 1$ and the upper complex is in the upstate. After the lower complex moves to the upstate, the sparsity remains at K (middle), however the upper complex is no longer minimally sparse. Let $x_{\min}$ denote the minimum loss of the child links in the upper complex **after** the move, and $m'_{\min}$ its associated multiplicity. Since $x_{\min} > 0$ and $m'_{\min} = 2 > 1$ sparsity can be reduced to $K + 1 - m'_{\min} = K - 1$ by moving the upper complex to the upstate.

There are two important observations to make from the above example. First, choosing the upstate always achieves minimal local sparsity. Second, choosing the upstate locally may also enable sparser states to be found in complexes higher in the tree. Together these motivate the following algorithm which defines a global solution based on the systematic exploitation of local sparsity with a preference for the upstate solution in case of local non-uniqueness. Recall that local complexes are centered on internal nodes at levels $l = 1, 2 \ldots H - 1$.

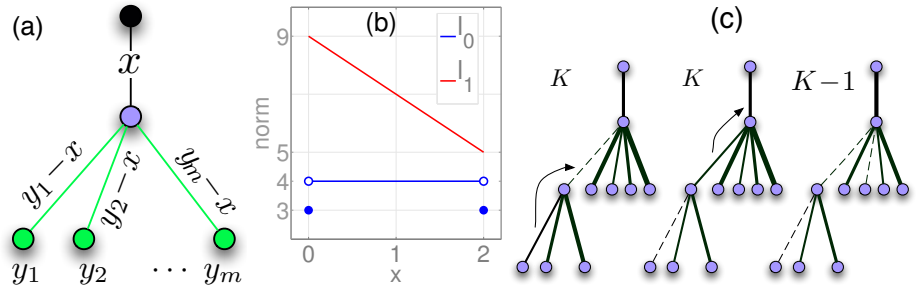

Fig. 3. (a) Ambiguity at a local complex, which could be centered on any internal node, illustrated by a simple tree. Links and paths are annotated with their loss values, obeying the general solution parameterized by x. Loss can be moved up or down subject to $x \in [0, y_{\min}]$. (b) ℓ_0 and ℓ_1 norms of solutions shown for a 3-receiver complex with observations $\boldsymbol{y} = [2, 3, 4]$. Loss can be moved up or down subject to $x \in [0, y_{\min} = 2]$. Both downstate and upstate solutions at $x = 0$ and $x = 2$ have the equal sparsity $K = m = 3$, but upstate achieves minimal ℓ_1. (c) An example of the coupling of two local complexes resulting in lower global ambiguity (link thickness proportion to loss, dashed links lossless). Left: initial configuration: lower complex in downstate, upper in upstate. Middle: system remains K-sparse when the lower complex changes state, but upper complex is no longer minimally sparse. Right: moving the upper complex into the upstate yields lower sparsity.

UpSparse Algorithm

Begin with an arbitrary feasible solution $\boldsymbol{x}$*. For all ambiguous local complexes at the deepest level select the up sparse state. Move up to the next level and repeat. Terminate at level 1.*

Function $UpSparse(\boldsymbol{x})$
% arbitrary initial solution $\boldsymbol{x}$
1: **for** $\ell = H - 1$ downto 1 **do**
2: **for all** nodes i at level ℓ **do**
3: $\boldsymbol{x} \leftarrow PutInUpState(i, \boldsymbol{x})$
4: **end for**
5: **end for**
6: **return** $\boldsymbol{x}$ % *UpSparse* Solution

Function $PutInUpState(i, \boldsymbol{x})$
% Put node i in upstate
1: $\delta \leftarrow \min_{j \in c(i)}\{x_j\}$
2: $x_i \leftarrow x_i + \delta$
3: **for all** nodes $j \in c(i)$ **do**
4: $x_j \leftarrow x_j - \delta$
5: **end for**
6: **return** $\boldsymbol{x}$ % Link i & children updated

We call the state of the loss vector tree after the application of the algorithm the *UpSparse solution*. An example of initial solution is the receiver solution $\boldsymbol{x}_I = 0, \boldsymbol{x}_R = \boldsymbol{y}$. Reading figure 2 from right to left provides an example of the algorithm in action. We now give its main properties. These are proved in [5].

UpSparse Properties

Let $\boldsymbol{x}^*$ be the output of UpSparse with input $\boldsymbol{x}$, then
Property 1. Each local complex in $\boldsymbol{x}^*$ is in upstate (by construction).

Property 2. *UpSparse uniqueness*: For fixed $\boldsymbol{y}$ and any feasible input $\boldsymbol{x}$, UpSparse outputs the same $\boldsymbol{x}^*(\boldsymbol{y})$, defined $\forall i \in V \backslash \{\mathcal{O}\}$ by:

$$x_i^* = \gamma_i \text{ if } f(i) = \mathcal{O}, \text{ else } x_i^* = \gamma_i - \gamma_{f(i)}, \text{ where } \gamma_i = \min \{ \boldsymbol{y}_{R(i)} \} \quad (3)$$

$\boldsymbol{y}_{R(i)}$ denotes the observations in subtree rooted at node i.
Property 3. *UpSparse solution has minimal ℓ_0 and ℓ_1 norms*: For any solution $\boldsymbol{x}$, $\|\boldsymbol{x}^*\|_0 \leq \|\boldsymbol{x}\|_0$ and $\|\boldsymbol{x}^*\|_1 \leq \|\boldsymbol{x}\|_1$.
Property 4. $\|\boldsymbol{x}^*\|_0 \leq m = |R|$.
Property 5. $\min \ell_1$ *uniqueness*: UpSparse solution $\boldsymbol{x}^*$ is the unique solution with $\min \ell_1$ norm.
Property 6. $\min \ell_0$ *uniqueness*: If each local complex in $\boldsymbol{x}^*$ is uniquely sparse, then $\boldsymbol{x}^*$ is the unique sparsest solution, otherwise not.
Corollary: When the number of lossy links at internal node i with degree g_i is at most $g_i - 2$ $\forall i$, then the sparsest solution is unique. This degree condition ensures that no internal node can be moved from upstate without increasing sparsity, and hence $\boldsymbol{x}^* = UpSparse(\boldsymbol{x}) = \boldsymbol{x}$.
Property 7. $\min \ell_1$*/UpSparse solution = true solution?*: If there exists at least one lossless child link at every internal node, UpSparse solution = true solution.

4.3 A sufficiency condition for non-uniqueness in graphs

Consider a measurement matrix A that defines a graph (instead of a tree) using paths from an underlying general network. Each column vector in A corresponds to a link in the graph. For any branch or internal node, let g^{in} (resp. g^{out}) denote its in-degree (resp. out-degree), that is the number of links covered by its incoming (resp. outgoing) paths. Then:
Lemma 1 *Let $K \geq \max\{g^{in}, g^{out}\}$ for some branch node. Then there exists a K-sparse non-negative vector $\boldsymbol{x}$ that is not the unique sparsest solution to $\boldsymbol{y}(\boldsymbol{x}) = A\boldsymbol{x}$.* (The proof is given in [5])

Lemma 1 gives a worst case bound relating the degree of networks and their amenability to sparse recovery. If networks have small degree, then sparsity minimization can fail even for small K if lossy links are concentrated at a branch point. For a binary tree, the bound for non-uniqueness given by lemma 1 is $K \geq 2$ since all branch nodes have $g^{in} = 1$ and $g^{out} = 2$. However property 6 above shows that sparsity minimization will still succeed for a binary tree for higher K provided the number of lossy links at each branch point is < 2.

4.4 The Noisy Problem

We briefly describe the noisy problem for completeness with details in [5]. The model defined in section 3 is based on knowing the mean loss observed at each receiver exactly. This assumption may fail in a number of ways, the most important of which is that, in practice, loss is estimated based on a finite number of observations, resulting in receivers seeing only an estimate $\widehat{\boldsymbol{y}}$ of the true path loss observation vector $\boldsymbol{y}$. We model this by associating a confidence interval $[y_j^\ell, y_j^u]$

for each receiver j with any $\boldsymbol{y} \in [\boldsymbol{y}^\ell, \boldsymbol{y}^u]$ (i.e. $\forall j, y_j \in [y_j^\ell, y_j^u]$) an equally possible observation vector. As before, we characterise the $\min \ell_0$ and $\min \ell_1$ solutions locally and globally within the tree. We observe that, unlike the noiseless case, the $\min \ell_1$ solution is no longer unique and need not always have the $\min \ell_0$ norm. We define **UpSparse**$^+$, a fast single-pass algorithm, which retrieves the $\min \ell_1$ solution out of all solutions with $\min \ell_0$.

5 Comparisons with CS Algorithms

Noiseless		Noisy	
$\min_x \ \|\|\boldsymbol{x}\|\|_0 : A\boldsymbol{x} = \boldsymbol{y}, x_i > 0 \ \forall i$	(4)	$\min_{\boldsymbol{x}} \ \frac{1}{2}\|\boldsymbol{y} - A\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$	(6)
$\min_x \ \|\|\boldsymbol{x}\|\|_1 : A\boldsymbol{x} = \boldsymbol{y}, x_i > 0 \ \forall i$	(5)	$\min_{\boldsymbol{x}} \ \|\boldsymbol{y} - A\boldsymbol{x}\|_1 + \lambda\|\boldsymbol{x}\|_1$	(7)
		$\min_{\boldsymbol{x}} \ \|\boldsymbol{y} - A\boldsymbol{x}\|_1 + \lambda\|\boldsymbol{x} - \boldsymbol{\mu}\|_1$	(8)

We now compare UpSparse to other optimisation methods from CS, of higher computational complexity, which can also find $\min \ell_0/\ell_1$ solutions in trees.

In the noiseless case, the optimal sparsity optimisation problem is (4) which is non-convex. In general such problems are NP-hard. In contrast, UpSparse provides a low cost single pass algorithm, which achieves global optimality through only $O(n)$ local operations, the number of links in the tree. In the ℓ_1 case, the optimisation problem is (5) which is not only convex but a linear program. Known as the basis pursuit formulation [13], it is used as a substitute for (4). Although its solution is straightforward using linear programming, UpSparse offers a low cost direct alternative which fully exploits the underlying tree topology.

The noisy problem has been tackled using the approaches (6-8). Eq.(6) is the basis pursuit denoising formulation and the unconstrained Lasso [17] formulation. Eq. (7) is used as an alternative to (6) as it is a linear program. It is used in [9]. Eq. (8) is proposed in [9] and used in [10] to choose a solution close to a prior $\boldsymbol{\mu}$. Compared to these approaches which require the introduction of a penalty term which is traded off with the ℓ_1 norm (used to approach ℓ_0) as well as a tradeoff parameter, UpSparse$^+$ uses ℓ_0 directly, supplemented by ℓ_1 when needed. A relevant special feature here of the tree problem is that **any** noisy observation $\widehat{\boldsymbol{y}}$ has a feasible solution (for e.g. the receiver solution). There is no need for regularisation in the sense of finding the closest feasible solution to $\widehat{\boldsymbol{y}}$.

6 Experiments

Uniqueness of $\min \ell_0$ *and effectiveness of* ℓ_1 *recovery in trees*: Figure 4(a) plots different probabilities of interest as a function of sparsity K by exhaustively looking at all $\binom{n}{K}$ links in trees. These are shown using two ternary trees with $n = 13$ ($m = 9$ leaves) and $n = 25$ ($m = 17$ leaves) links for $1 \leq K \leq m$. The red curves show the probability that the sparsest solution is unique. The blue curves show the probability that the $\min \ell_1$ solution is the true underlying solution.

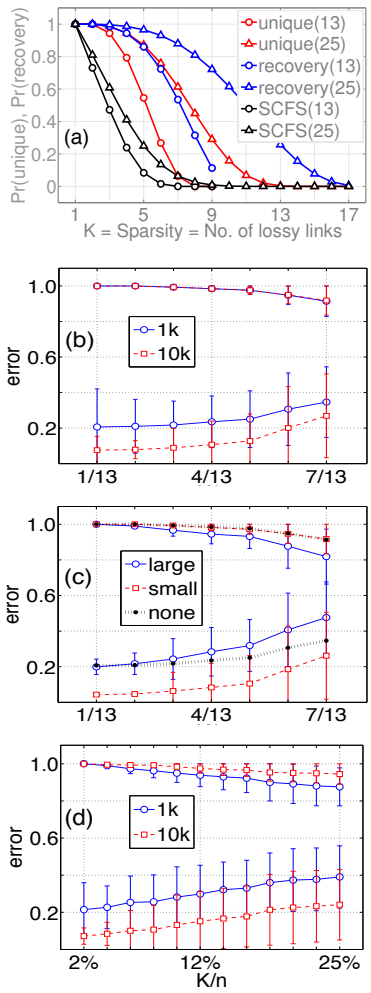

Fig. 4. Top to bottom: (a) Probabilities of uniqueness and recovery of true solutions. (b-d): Effect of measurement noise for trees as a function of K/n(sparsity/number of links). Top curves show the success rate e_0 and bottom curves show relative error e_2 (b, c): Results for UpSparse and UpSparse$^+$ solutions respectively for a ternary tree with $n = 13$ links, and (d): Results for realistic trees from AT&T network.

Property 6 of section 4.2 gives the condition for uniqueness of sparsest solution and 7 gives the condition that $\min \ell_1$ solution = true solution. For $K \leq 2$, $\min \ell_0$ is guaranteed to be unique since for all internal nodes, the node degree $g = 4$ (3 children, 1 parent) and $K \leq g - 2$ holds. For $K > 2$, the probabilities gradually decay with increasing number of lossy links. For $K > 2$, a vector of sparsity K need not be unique in general (lemma 1). It is unique however if corollary of property 6 is satisfied. We see that ℓ_1 minimization can recover the true solution even when the sparsest solution is not unique. Since ℓ_1 picks one of the sparsest solutions that is in upstate, when the true solution is the upstate solution, ℓ_1 minimization recovers it. It is clear that UpSparse or any ℓ_1 minimization algorithm effectively recovers the true loss hotspots in a tree, provided that only few of them exist. Next, we compare UpSparse to SCFS (smallest consistent failure set) algorithm [6], which recovers the sparsest link binary vector (each link either good/lossless or bad/lossy) given the path binary vector (each path either good or bad). The black curves show the probability that SCFS recovers the true link binary vector *i.e.* the locations of all bad/lossy links. We see that SCFS has a lower success rate than UpSparse (blue curves) even though UpSparse recovers both the locations as well as loss rates of all lossy links. The binary approach yields higher ambiguity than the loss approach. For example, at a branch node, if one of the child links and the parent link are both bad/lossy, SCFS will report only the parent as bad. However UpSparse will report both links as lossy.

Effect of measurement noise: Next we conduct probing experiments where instead of the true path probabilities only an estimate is available, based on a fixed number of probes. We compute the minimal norm solutions and plot errors as a function of both sparsity and increasing number of probes. We assume a

scenario where we have no prior knowledge of where the lossy links may lie. Thus we pick K links uniformly at random. For each of these, loss is set uniformly at random from $1 - 10\%$. For the remaining links, loss is set to 0. Using $\boldsymbol{x}$, we simulate the passage of probes on each path and derive the noisy observation $\widehat{\boldsymbol{y}}$ and its confidence intervals $\widehat{\boldsymbol{y}}^{\ell}$, and $\widehat{\boldsymbol{y}}^{u}$. These are used by UpSparse and UpSparse^{+} to yield $\widehat{\boldsymbol{x}}$. Finally we recover the true loss estimates $\widehat{\mathbf{b}} = \mathcal{L}^{-1}(\widehat{\boldsymbol{x}})$.

We compute two quantities: (i) relative ℓ_2 error $e_2 = \|\mathbf{b} - \widehat{\mathbf{b}}\|_2 / \|\mathbf{b}\|_2$ and (ii) Success rate e_0 which is the number of common lossy links between $\mathbf{b}$ and $\widehat{\mathbf{b}}$ normalised by $\|\mathbf{b}\|_0$. e_0 attempts to measure the effectiveness of UpSparse in identifying the correct locations of lossy links.

1) UpSparse Solutions: Figure 4(b) shows benchmark results using a ternary tree with $n = 13$ links and height 3. The top curves show the success rate e_0 and bottom ones show the error e_2 averaged over 100 repetitions for 1k and 10k probes when UpSparse is given the noisy $\widehat{\boldsymbol{y}}$ in each repetition. We see that even with 1k probes (blue curve), UpSparse identifies the correct locations of majority of the lossy links. As probes increase, $\widehat{\boldsymbol{y}}$ approaches $\boldsymbol{y}$ and e_2 decreases.

2) UpSparse^{+} Solutions: Figure 4(c) shows results for UpSparse^{+} using 1k probes for intervals of small and large sizes. In each repetition, UpSparse^{+} is given intervals $\widehat{\boldsymbol{y}}^{\ell}$, and $\widehat{\boldsymbol{y}}^{u}$ that contain the true $\boldsymbol{y}$. When interval sizes are large, UpSparse^{+} can find solutions which are even sparser than $\boldsymbol{x}$ resulting in higher error. As intervals get narrower, UpSparse^{+} gives better results. The curve in the centre shows results when the actual noisy observation is used by UpSparse.

3) Large Realistic trees: Finally 4(d) shows results for real tree topologies cut out from the publicly available router level map of the AT&T network obtained by Rocketfuel [18], with about 48 links on average per tree. The figure shows results for solutions computed using UpSparse^{+} for 1k and 10k probes when $\widehat{\boldsymbol{y}}^{\ell}$, and $\widehat{\boldsymbol{y}}^{u}$ are t-distributed intervals for 90% confidence, centered around $\widehat{\boldsymbol{y}}$. We see that e_2 increases as expected as large trees will need more probes to get accurate path probabilities. However success rate e_0 remains high implying that minimal norm solutions identify the correct locations of majority of the lossy links.

7 Conclusion

The sparsity principle has been mostly used as a black box in loss tomography without detailed characterization of conditions under which minimal norm solutions are unique or recover the true underlying solution. These conditions are important in practice as network operators wish to know when sparsity could be used to accurately localise hotspots using few monitoring points.

In this work, we study the problem of loss hotspot localization in tree topologies (e.g. server-based measurements) using the principle of sparsity. We derive explicit solutions and fast algorithms for both $\min \ell_0$ and $\min \ell_1$ norms that give deep insight into the nature of sparsity in trees. We provide conditions under which minimal norm solutions are unique and when they recover the true underlying solution. We show that when lossy links are well separated, sparse solutions remain unique in many cases. We conduct experiments to measure the

ability of the minimally sparse solution to approach the actual sparse solutions in practice. We see that minimal norm solutions can identify the locations of most lossy links, however as their number increases it becomes much harder to identify true loss rates. We also observe that minimally sparse link loss solutions can localize hotspots better than minimally sparse link binary solutions used in prior work.

Future work will extend our work to graphs, study recovery conditions for the binary performance problem, and test our results with real measurements.

References

1. Padmanabhan, V.N., Qiu, L., Wang, H.J.: Server-based inference of internet link lossiness. In: IEEE INFOCOM. (Mar 2003) 145 – 155
2. Candes, E., Tao, T.: Decoding by linear programming. IEEE Transactions on Information Theory **51**(12) (2005) 4203 – 4215
3. Candés, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory **52**(2) (2006) 489–509
4. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM Journal on Computing **24**(2) (1995) 227–234
5. Arya, V., Veitch, D.: Sparsity without the complexity: Loss localisation using tree measurements. CoRR **abs/1108.1377** (2011) http://arxiv.org/abs/1108.1377.
6. Duffield, N.G.: Network tomography of binary network performance characteristics. IEEE Transactions on Information Theory **52**(12) (2006) 5373–5388
7. Chen, Y., Bindel, D., Song, H., Katz, R.H.: An algebraic approach to practical and scalable overlay network monitoring. In: ACM SIGCOMM. (Aug 2004) 55–66
8. Chua, D., Kolaczyk, E., Crovella, M.: Network Kriging. IEEE Journal on Selected Areas in Communications **24**(12) (2006) 2263–2272
9. Song, H.H., Qiu, L., Zhang, Y.: Netquest: a flexible framework for large-scale network measurement. IEEE/ACM Trans. Netw. **17**(1) (2009) 106–119
10. Ghita, D., Nguyen, H., Kurant, M., Argyraki, K., Thiran, P.: Netscope: practical network loss tomography. In: IEEE INFOCOM. (Mar 2010) 1262–1270
11. Adams, A., Bu, T., Caceres, T., Duffield, N.G., Friedman, T., Horowitz, J., Lo Presti, F., Moon, S.B., Paxson, V., Towsley, D.: The use of End-to-end Multicast Measurements for Characterising Internal Network Behavior. IEEE Communications Magazine **38**(5) (May 2000) 152–158
12. Donoho, D.L.: Compressed sensing. IEEE Transactions on Information Theory **52**(4) (2006) 1289–1306
13. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM Rev. **43**(1) (2001) 129–159
14. Firooz, M.H., Roy, S.: Network tomography via compressed sensing. In: IEEE GLOBECOM. (Dec 2010) 1–5
15. Indyk, P.: Sparse recovery using sparse random matrices. In: LATIN. (Apr 2010)
16. Xu, W., Mallada, E., Tang, A.: Compressive sensing over graphs. In: IEEE INFOCOM. (Mar 2011) 2087–2095
17. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B Methodological **58** (1996) 267–288
18. Spring, N., Mahajan, R., Wetherall, D., Anderson, T.: Measuring ISP topologies with Rocketfuel. IEEE/ACM Trans. Netw. **12**(1) (2004) 2–16