Prediction of Bandwidth and Additive Metrics for Large Scale Network Tomography

Microsoft Research Technical Report MSR-TR-2016-57

Christoffer A. Rødbro
Microsoft Skype
Stockholm, Sweden
crodbro@microsoft.com

Philip A. Chou
Microsoft Research
Redmond, WA
pachou@microsoft.com

Ürün Dogan
Microsoft Research
Cambridge, UK
udogan@microsoft.com

ABSTRACT

For real time communication services over the Internet, it is important to be able to predict in advance the quality of a call before relaying it over a particular path. In this paper we show how to predict the distribution of the end-to-end bandwidth, latency, jitter, and loss of a call from an arbitrary user X to an arbitrary user Y through particular components of the Internet, given a dataset of millions of calls among other users. This work is the first to infer component bandwidth distributions for an arbitrary network topology, and the first to infer component bandwidth and additive metric distributions for an arbitrary network topology at large scale. On a dataset of over seven million global Skype calls, we demonstrate significant performance improvement compared to a baseline approach used in today's commercial systems.

1. INTRODUCTION

Real time "over-the-top" (OTT) audio and video communication services such as Skype, Hangouts, Facetime, Viber, and others are promising to replace traditional telephony. Already, Skype alone carries multiple billions of audio and video minutes per day [1] with double-digit growth [2]. However, OTT services today are subject to frequent disruptions due to inter-domain routing, whether caused by congestion deep in the network [3], BGP convergence issues [4], or business relationships [5]. To improve OTT services, it would be extremely helpful to have a real time performance map of the Internet. By knowing where the problem areas are, for example, OTT services could largely circumvent them by relaying around them [6, 7, 8].

In principle, network tomography could be used to obtain a real time performance map of the Internet from end-to-end measurements. This approach is especially attractive since these OTT services can already collect end-to-end performance data from their clients on every call, based on passive measurements of in-band packet transmission. For large OTT services, such big data could be used to infer performance in the Internet on a large scale.

Unfortunately, even with such massive amounts of data, accurate topology inference and up-to-the-minute estimates of the state of every link in the Internet is probably not possible. Fortunately, it is probably not necessary either. In this paper, we take the point of view that the purpose of network tomography is accurate *prediction* of end-to-end performance along any given path, in particular paths that have not been observed much in recent data, if ever. While we do infer performance distributions across network-internal components, these components are but part of our network *model*. We train the parameters of the model on a large dataset, and validate the model on a separate test set in terms of prediction error or cross entropy of the test set. This approach, which is typical in machine learning, meshes well with the reality that in large scale network tomography, it will never be possible to validate against the ground truth, yet prediction accuracy is simple to check and compare.

In this context, our primary contribution is the introduction of novel methods for inference of the parameters of network components such as bandwidth, round trip time, jitter, and packet loss distributions from end-to-end measurements, suitable for inference on a large scale. We demonstrate our methods on a dataset of seven million Skype calls, one-third of which are held out for independent validation.

1.1 Related Work

There has been a large body of work on Network Tomography since the original work of Vardi [9]. While some work, including Vardi's, has focused on the problem of inferring network externals (traffic matrices) from network internals (link metrics), most of the work has focused on the converse problem of inferring network internals (link metrics and network topologies) from network externals (end-to-end measurements). In the latter category, Cáceres et al. [10] showed how to infer loss probabilities on each link in a multicast tree by actively injecting probe packets, and proved consistency of their estimates. Lo Presti et al. [11] ex-

tended the approach to delay distributions, where the distributions were represented by probability mass functions of clipped and quantized delays. Duffield and Lo Presti [12] extended the approach to delay variance, or jitter. Coates and Nowak [13] and then Duffield et al. [14] employed bursts of unicast packets along a tree rather than multicast, motivated by the limited deployment of IP multicast. In these early works, maximum likelihood or pseudo-likelihood, computed by the expectation-maximization algorithm, were the usual approaches to estimation. The tree structure, combined with multicast or strongly correlated unicast packets, made it possible to estimate with consistency, i.e., *identify*, the link metrics from end-to-end measurements (a property shared with phylogenetic trees [15]). Overviews of these early works can be found in [16, 17].

Passive measurement using existing flows, that is, without active injection of probe packets, was first performed in [18], while topologies more general than trees began to be examined in [19, 20]. Among the latter, Y. Chen et al. [20] showed how to infer link loss rates from end-to-end measurements on an arbitrary set of point-to-point paths, which we will here call a mesh. They also introduced using the log transformation of the success probabilities and solving a set of linear equations, calling out the problem of identifiability only up to a subspace. Later, A. Chen et al. [21] and Thoppe [22] solved the linear equations for additive metrics using characteristic functions.

The move to general topologies and passive measurements caused identifiability to become a key concern, as correlated measurements in a tree structure were no longer possible. Identifiability in mesh topologies with additive metrics was studied rigorously in [23]–[29]. However, Duffield [30] and others [31]–[35] realized that regardless of the absence of strict identifiability, simple and practical network tomography could be performed based on the sparsity of congestion events. This was formalized by the use of compressive sensing in network tomography [36]–[42].

Work on bandwidth tomography using end-to-end bandwidth measurements has been rare, perhaps because bandwidth is generally not strictly identifiable. Baralis et al. [43] and Dichev et al. [44] performed coarse-grained bandwidth estimation based on clustering measurements between peer-to-peer clients including BitTorrent. Dinwoodie [45] addressed the problem of bandwidth tomography on multicast trees requiring active probing.

A prerequisite to bandwidth tomography is the ability to observe end-to-end bandwidths over paths. Such an end-to-end bandwidth observation is itself an estimate of the bandwidth along the path. For this estimation problem there is an even richer literature than network tomography, whether it is for capacity estimation (e.g., [46]), available bandwidth estimation (e.g., [47]), TCP friendly rate control (e.g., [48]), or the rate control for streaming media (e.g., [49]). The latter work is most relevant to our work, as we use the client's bandwidth estimator for our end-to-end bandwidth observations.

The most recent trend in network tomography is using network coding [50]–[59]. However, network coding is even less likely than multicast to be deployed at the network internals. In general, we find very little evidence of practical network tomography on a large scale.

1.2 Our Contributions

Our main contributions are new techniques for solving for the distributions of bandwidth and additive metrics on the internal links (or *components*) of a network from passive end-to-end measurements, suitable for large scale tomography of a network with arbitrary (mesh) topology. Additive metrics include (after transformation if necessary) delay, jitter, and loss. For both bandwidth and additive metrics, we derive expressions for the distributions of the end-to-end measurements as a function of the distributions on the internal network components. For bandwidth, the expression is based on order statistics, while for additive metrics, it is based on convolution. We then explicitly optimize the component distributions using modern optimization methods [60] to maximize the likelihood on a training set. For the additive metrics, the Fast Fourier Transform plays an important computational role.

Unlike all the early references ([9]–[61]), our approach is applicable to passive point-to-point measurements over arbitrary network topologies. These characteristics are crucial for our application, in which the measurements made at existing clients in peer-to-peer calls are used to infer network internal characteristics across the Internet. Unlike [19] and [20], which are applicable to this scenario but estimate only loss, we estimate bandwidth and other additive metrics. Most similar to our bandwidth technique is work by Dinwoodie [45]. However [45] requires a tree and active probing, and is therefore fundamentally inapplicable to our scenario. Most similar to our additive metric technique is work by Chen et al. [21] and Thoppe [22], in that they use moment generating functions, which is similar to our use of Fourier transforms. However, they restrict their topology to trees. Moreover, Chen et al. minimize squared error in the moment generating function domain and Thorpe matches moments, while we maximize likelihood directly. We also use the FFT, making our method extremely fast. Tsang et al. [62] use the FFT, but its use is explicitly tied to computing convolutions in a message-passing algorithm, which assumes a tree topology. In contrast, we introduce new techniques for estimating bandwidth and additive metrics that apply to point-to-point measurements over arbitrary network topologies.

Our second contribution is scale. We demonstrate our techniques at large scale on a dataset of over seven million calls, using a network model containing over 22,000 components and over four billion possible paths, representing the connectivity of users among 11,000 Internet service providers. The previously largest reported network tomography experiments we are aware of are the PlanetLab experiments in [31], with around 5000 components and a similar number of paths.

At a large scale, the ground truth is unknowable. Therefore our emphasis is on modeling and accurate prediction. That is, we model the network and its parameters in such a way that it can accurately *predict* the performance on paths or combination of components that have never been observed before, or have been observed with only few samples. We do not solve the problem of strict identifiability. In fact, the highest link bandwidth is almost never identifiable, being masked by lower bandwidths. However, since prediction is the goal, identifying the highest link bandwidth is almost always irrelevant. This viewpoint is perhaps our third, small contribution, as it changes the conversation from the problem of identifiability to the problem of predictability.

1.3 Organization of the Paper

Section 2 details our approach. Section 3 describes the dataset of calls and the network topology we use in our experiments. Section 4 provides our experimental results. Section 5 concludes.

2. METHODS

In this section we shall first take a look at how we model end-to-end quality of service (QoS) observations, while defining our notation. We shall then derive the likelihood functions for four QoS metrics: bandwidth, round trip time (RTT), jitter, and loss, treating separately the additive (RTT, jitter, and loss) and non-additive (bandwidth) metrics.

2.1 Generic Network and QoS Model

We adopt a simple and generic network model, as in [63]. Specifically, we model a network as a (usually large) collection of *components*, $\mathcal{A}$. We shall assign a precise interpretation to the components in the network model we use in Section 3, but for now think of the components as logically disjunct parts of the network, such as individual routers, links, and servers in a fine-grained network model, or ASNs or even countries in a coarse-grained network model. Now, across the network we make end-to-end observations $\mathbf{y}$ of one of the four QoS metrics and we refer to the ith end-to-end observation as y_i. To avoid unnecessary notational complication we will not specifically indicate which QoS metric is in question as that will be obvious from the context. The end-to-end observations involve a (usually small) subset of $\mathcal{A}$, which we call a *path*. We denote by $\mathcal{A}_i$ the components of the path on which y_i was observed; it is assumed to be known for each observation y_i.

The end-to-end QoS observations y_i are modeled as realizations of independent random variables Y_i generated as follows. Along the path $\mathcal{A}_i$, each network component $j \in \mathcal{A}_i$ produces a QoS contribution x_{ij}, which is a realization of a random variable X_{ij} drawn independently from a probability distribution $p_j(x)$ associated with component j. Then, for the non-additive QoS metric (bandwidth),

$$y_i = \min_{j \in \mathcal{A}_i} x_{ij}, \tag{1}$$

while for the additive QoS metrics (RTT, jitter, and loss),

$$y_i = \sum_{j \in \mathcal{A}_i} x_{ij}. \tag{2}$$

While RTT is clearly additive, jitter and loss require transformation to be additive. Specifically, the square of the jitter is additive if jitter is measured as the standard deviation of the latency, since the variance of a sum of independent random variables is the sum of their variances. (However, jitter is often measured as the mean absolute deviation of the latency [64, appendix A.8], which is only approximately additive.) Similarly, the log complement of the packet loss is additive, since the end-to-end packet loss $L_i = 1 - \prod_j (1 - L_j)$ becomes additive after a simple transformation:

$$\log(1 - L_i) = \sum_j \log(1 - L_j). \tag{3}$$

Thus, we can treat RTT, jitter, and loss as being (at least approximately) additive, for which the corresponding likelihood function will be derived in Section 2.2. However, there is no reasonable transformation that makes bandwidth additive, because bandwidth is determined by the narrowest bottleneck on the path. This poses a more difficult problem, and the likelihood function will be derived in Section 2.3.

Finally, we model the component distributions $p_j(x)$ as an element of some family of distributions. For our purposes, we find that the family of continuous distributions given by densities that are piecewise constant within fixed bins is sufficient. Specifically, we model $p_j(x)$ by a vector of probabilities $\mathbf{p}_j$ and a set of K uniform intervals; the details will be described later. As $\mathbf{p}_j$ represents a probability mass function (pmf), we require that it sums to unity and that each element is a valid probability:

$$\mathbf{1}^T \mathbf{p}_j = 1 \tag{4}$$

$$0 \preceq \mathbf{p}_j \preceq 1 \tag{5}$$

The complete set of $\mathbf{p}_j$ for all j we refer to as $\mathcal{P}$ which thus constitutes the parameters of our model.

Our objective is to use a training set of end-to-end QoS observations $\mathbf{y}$ to find values for the parameters $\mathcal{P}$ that allow the model to best predict the end-to-end QoS distributions on paths not observed much, if at all, in the training set. Recovering the underlying "true" component QoS distributions may be sufficient, but is not necessary, to achieve our objective.

To achieve our objective, our approach is to find the parameter values $\mathcal{P}$ that maximize the likelihood of the training set $\mathbf{y}$ given $\mathcal{P}$. This likelihood is derived in the next two subsections.

2.2 Likelihood Function for Additive Metrics

For additive metrics, the end-to-end pdf is given by the convolution of the pdfs for each of the N components on the path:

$$p(y_i) = p_{k_1}(x) * p_{k_2}(x) * ... * p_{k_N}(x), \quad \mathcal{A}_i = \{k_1, k_2, ..., k_N\} \tag{6}$$

This expression is a little hard to work with until we replace the generic pdf's by their discretized counterparts and calculate the convolutions via the discrete Fourier transform $\mathbf{F}$,

$$\mathbf{F}\tilde{\mathbf{y}}_i|_{\mathcal{P}} = \prod_{j \in \mathcal{A}_i} \mathbf{F}\tilde{\mathbf{p}}_j, \tag{7}$$

where the product is taken vector-elementwise, $\tilde{\mathbf{y}}_i|_{\mathcal{P}}$ is the pmf of the end-to-end observation Y_i given the model parameters $\mathcal{P}$, and $\tilde{\mathbf{p}}_j$ is the pmf $\mathbf{p}_j$ zero-padded to avoid aliasing. With N components on the path, $\tilde{\mathbf{p}}_j$ must be zero-padded to length $N(K-1)+1$, assuming all discretizations $p_j(x) \to \mathbf{p}_j$ use equally spaced bins starting at 0. Thus, we can recover $p(y_i|\mathcal{P})$ by taking the inverse Fourier transform and pinning $\tilde{\mathbf{y}}_i|_{\mathcal{P}}$:

$$p(y_i|\mathcal{P}) = \left(\mathbf{F}^H \prod_{j \in \mathcal{A}_i} \mathbf{F}\tilde{\mathbf{p}}_j \right)_{(b_i)} = \mathbf{f}_{b_i}^H \prod_{j \in \mathcal{A}_i} \mathbf{F}\tilde{\mathbf{p}}_j, \tag{8}$$

where b_i is the bin corresponding to observation y_i and $\mathbf{f}_{b_i}$ is the corresponding Fourier transform basis vector. Thus, the log-likelihood of the entire dataset $\mathbf{y}$ is

$$\log\left(p(\mathbf{y}|\mathcal{P})\right) = \log \prod_i p(y_i|\mathcal{P}) = \sum_i \log \left(\mathbf{f}_{b_i}^H \prod_{j \in \mathcal{A}_i} \mathbf{F}\tilde{\mathbf{p}}_j \right). \tag{9}$$

This expression is simple to evaluate (the inner products $\mathbf{F}\tilde{\mathbf{p}}_j$ via FFT) and can easily be differentiated in $\mathbf{p}_j$ and can therefore serve as a basis for maximum likelihood optimization together with the constraints (4) and (5). Unfortunately, it is not convex but as we shall see in a later section, empirical results show that we are indeed able to find a usable maximum for it anyway.

2.2.1 Toy Example

In order to investigate the fundamental behavior and limitations of the proposed QoS modeling, we will take a look at a toy example with only a few network components. For each call, each component is configured to introduce a round-trip time delay that is drawn from a Rayleigh distribution with mean given by the component index times 100 ms; for example component number 2 adds 200 ms of RTT on average. Each call traverses two randomly chosen components and we use a total of 5000 calls. The 8 histogram bins are evenly spaced between 0 and 1200 ms. Figure 1 shows the underlying as well as estimated pmfs using the proposed max-likelihood based method for 2, 3 and 4 network components. We see that for two components, the estimated pdf's appear to be swapped; the reason is that this scenario is underdetermined so that the likelihood function reaches its maximum for any solution where the component pdf convolution matches the pdf of the end-to-end observations. Once we add a 3rd and 4th component we see that the pdfs are recovered well.

2.3 Likelihood Function for Bandwidth

Deriving a similar expression for bandwidth is somewhat harder. Here, we will keep much of the derivation general, as we do not need to resort to discretization immediately. We remember that the end-to-end bandwidth is given by the smallest component according to (1). Therefore we first need the probability that component j delivers a bandwidth high enough to yield y_i:

$$P(X_j \geq y_i) = \int_{y_i}^{\infty} p_j(x)dx. \tag{10}$$

Now the probability that all components deliver a high enough bandwidth is:

$$P(Y_i \geq y_i) = \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x)dx. \tag{11}$$

Now we get back to the likelihood function by:

$$\begin{aligned} p_i(y_i) &= \frac{d}{dy_i} P(Y_i \leq y_i) = \frac{d}{dy_i}\left(1 - P(Y_i \geq y_i)\right) \\ &= -\frac{d}{dy_i} \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x)dx. \end{aligned} \tag{12}$$

For the whole set of observations $\mathbf{y}$ the likelihood thus becomes:

$$p(\mathbf{y}) = \prod_i -\frac{d}{dy_i} \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x)dx. \tag{13}$$

As we are interested in finding the underlying density functions $p_j(x)$ that maximize $p(\mathbf{y})$ we can take the logarithm on both sides to arrive at the log-likelihood func-

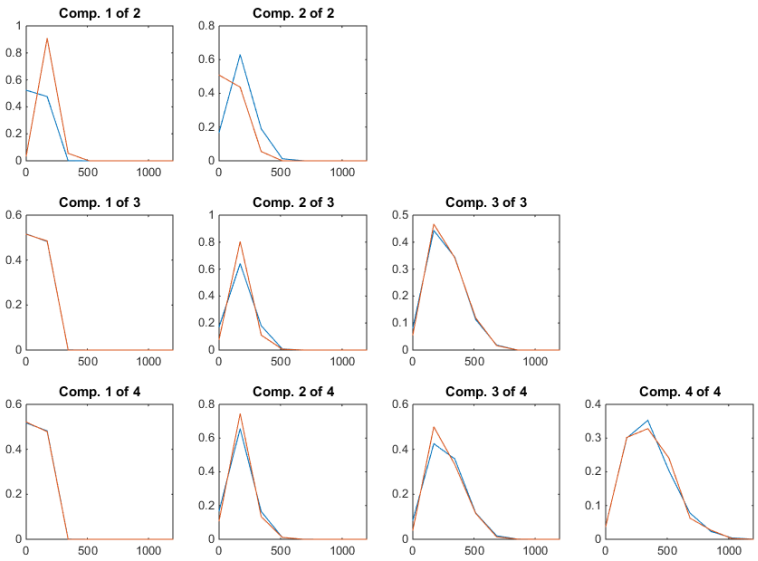

Figure 1: Toy example of component RTT pdf recovery using Rayleigh data. The true pdf is shown in blue, the estimated in red.

tion

$$\log p(\mathbf{y}) = \sum_i \log \left(-\frac{d}{dy_i} \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx \right). \quad (14)$$

This term looks a bit intimidating until we note that

$$\frac{d}{dy_i} \log \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx = \frac{\frac{d}{dy_i} \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx}{\prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx}, \quad (15)$$

and thus we can rewrite

$$\begin{aligned}
&\log p(\mathbf{y}) \\
&= \sum_i \log \left(\prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx \cdot -\frac{d}{dy_i} \log \prod_{j \in \mathcal{A}_i} \int_{y_i}^{\infty} p_j(x) dx \right) \\
&= \sum_i \Bigg[\sum_{j \in \mathcal{A}_i} \log \int_{y_i}^{\infty} p_j(x) dx \\
&\quad + \log \sum_{j \in \mathcal{A}_i} -\frac{d}{dy_i} \log \int_{y_i}^{\infty} p_j(x) dx \Bigg] \\
&= \sum_i \sum_{j \in \mathcal{A}_i} \log \int_{y_i}^{\infty} p_j(x) dx + \sum_i \log \sum_{j \in \mathcal{A}_i} \frac{p_j(y_i)}{\int_{y_i}^{\infty} p_j(x) dx}.
\end{aligned} \quad (16)$$

We now have a generic utility to optimize for given any choice of distribution $p_j(x)$. For example, if we plug in any Weibull-class distribution with fixed shape metric $k > 0$ as was also used in [45],

$$\begin{aligned}
p_j(y_i|\theta_j) &= k\theta_j^k y_i^{k-1} e^{-\theta_j^k y_i^k}, \qquad \theta_j > 0 \quad (17) \\
\int_{y_i}^{\infty} p_j(x|\theta_j) dx &= e^{-\theta_j^k y_i^k} \quad (18)
\end{aligned}$$

we arrive at:

$$\log p(\mathbf{y}|\theta_1 \dots \theta_J) = -\sum_i \sum_{j \in \mathcal{A}_i} \theta_j^k y_i^k + \sum_i \log \sum_{j \in \mathcal{A}_i} k y_i^{k-1} \theta_j^k. \quad (19)$$

We note that for fixed k this expression in concave in the set of $\theta_\cdot^k$s by inspection and the maximum likelihood problem is therefore easily solvable under the simple $\theta_j > 0$ constraint.

However, we do not want to fit Weibull distributions just because it leads to simple optimization criteria; we can make no assumptions regarding the distributions and therefore want parameterless fits. As in the previous subsection, we discretize the observed bandwidths into bins and denote the appropriate bin for observation y_i by b_i and the corresponding bin center by $\tilde{y}(b_i)$ (that is, a quantized y_i) and can write (16) as:

$$\begin{aligned}
&\log p(\mathbf{y}|\mathcal{P}) \\
&= \sum_i \sum_{j \in \mathcal{A}_i} \log \left(\alpha_i w(b_i) p_j(b_i) + \sum_{k=b_i+1}^{K} w(k) p_j(k) \right) \\
&+ \sum_i \log \sum_{j \in \mathcal{A}_i} \frac{p_j(b_i)}{\alpha_i w(b_i) p_j(b_i) + \sum_{k=b_i+1}^{K} w(k) p_j(k)},
\end{aligned} \quad (20)$$

where K is the number of bins, $w(k)$ is the width of the kth bin, and

$$\alpha_i = (\tilde{y}(b_i) - y_i)/w(b_i) + 1/2 \tag{21}$$

is the fraction of bin b_i above y_i. Defining $p_{jk} = w(k)p_j(k)$ to be the probability of the kth bin of jth component, (20) reduces to

$$\log p(\mathbf{y}|\mathcal{P}) = \sum_i \sum_{j \in \mathcal{A}_i} \log \left(\alpha_i p_{jb_i} + \sum_{k=b_i+1}^{K} p_{jk} \right) + \sum_i \log \sum_{j \in \mathcal{A}_i} \frac{p_{jb_i}/w(b_i)}{\alpha_i p_{jb_i} + \sum_{k=b_i+1}^{K} p_{jk}}, \tag{22}$$

which can be vectorized as

$$\log p(\mathbf{y}|\mathcal{P}) = \sum_i \sum_{j \in \mathcal{A}_i} \log \mathbf{w}_i^T \mathbf{p}_j + \sum_i \log \sum_{j \in \mathcal{A}_i} \frac{\mathbf{a}_i^T \mathbf{p}_j}{\mathbf{w}_i^T \mathbf{p}_j}, \tag{23}$$

where $\mathbf{a}_i$ is $1/w(b_i)$ at index b_i and is 0 otherwise, and $\mathbf{w}_i$ is α_i at index b_i, 0 below b_i, and 1 above. This is a differentiable likelihood criteria that can serve as a basis for maximum likelihood estimation together with the constraints (4) and (5). In the practical implementation we approximate (21) by $\alpha_i = 1/2$ which significantly reduces the complexity of evaluation of (23) and its derivatives because $\mathbf{w}_i$ and $\mathbf{a}_i$ now take on only K different values allowing precomputation of the inner summation terms. Just as in the additive metric case (23) is not concave so we shall rely on empirical investigations to illustrate the efficacy of the optimization.

Finally, we note that the only precondition for the derivation was (1), meaning that (23) is also valid for any non-decreasing function of bandwidth. Thus, for real-world data (Section 4) we prefer working with log-bandwidth as that is better correlated with quality of experience.

2.3.1 Toy Example

In order to investigate the fundamental behavior and limitations of the proposed bandwidth modeling, we redo the experiment of Section 2.2.1 but for bandwidth instead of RTT. Serving bandwidths are drawn from Rayleigh distributions with means given by the component index times 100 kbps. Again, each call traverses two randomly chosen components and we use a total of 5000 calls. The 8 histogram bins are centered at 100, 200,...800 kbps. Figure 2 shows the underlying as well as estimated pdfs using the proposed max-likelihood based method for 2, 3 and 4 network components. Again we see that for two components, the estimated pdfs are identical due to the system being underdetermined. However, once we add a 3rd component we see that the pdfs for the 100 and 200 kbps components are recovered well, whereas the 300 kbps component is inaccurate and underestimated. The pattern repeats itself when we add a 4th 400 kbps component: the 300 kbps component is now well recovered but not the 400 kbps. This is not surprising: for most calls that traverse the highest bandwidth component the observed end-to-end bandwidth is determined by another component and therefore any pdf that is "high enough" explains the observed data equally well.

2.4 Tikhonov Regularization

Empirically we find it advantageous to add differential Tikhonov regularization terms to the objectivity functions:

$$T(\mathcal{P}) = \alpha_T \sum_j \|\mathbf{D}\mathbf{p}_j\|_2 \tag{24}$$

where $\mathbf{D}$ is the differential operator and α_T is a constant scaling factor. This trick leads to slight smoothing of the estimated pdf's, especially for components j that are only observed a few times in the entire observation set. A value of $\alpha_T = -5$ is found to be suitable for all scenarios (including the toy examples above).

2.5 Likelihood Optimization Details

At this point, we can state our problem as: maximize the log-likelihood function (9) or (23) plus Tikhonov regularization (24) under the linear equality- and inequality constraints (4) and (5). To this end, we rely on the Log-barrier method of [65, chapter 10.2] although our objectivity functions are not concave, and as such, a globally optimal solution is not guaranteed. The method requires computation of the gradient and Hessians of the log-likelihood functions, which is a tedious but straightforward task. We empirically find the most computationally intensive task to be the calculation of the elements of the Hessian matrix. Therefore, we calculate only its diagonal elements, greatly reducing the memory requirements and computational complexity of each iteration at the price of higher optimization loop iteration counts. The Log-barrier method was implemented in Matlab with the objectivity, gradient and Hessian calculations offloaded to multithreaded C++ via mex-interfaces. For the Skype dataset and chosen topology (see Section 3) with 4.8M training samples the optimization runs in less than two hours for additive metrics and less than three hours for bandwidth on an dual-core Intel i7 based laptop. While these processing times are fast enough to allow daily processing of similar datasets speedups can easily be achieved by distributing the Hessian and gradient accumulation over more cores.

Empirically and aside from the toy examples, we find our methods to be quite insensitive to parameter initialization. For bandwidth, we simply initialize the pmfs as uniform distributions, $\forall j, k : p_j(k) = 1/K$. For additive metrics, we initialize so that for most paths the component convolution match the overall training set

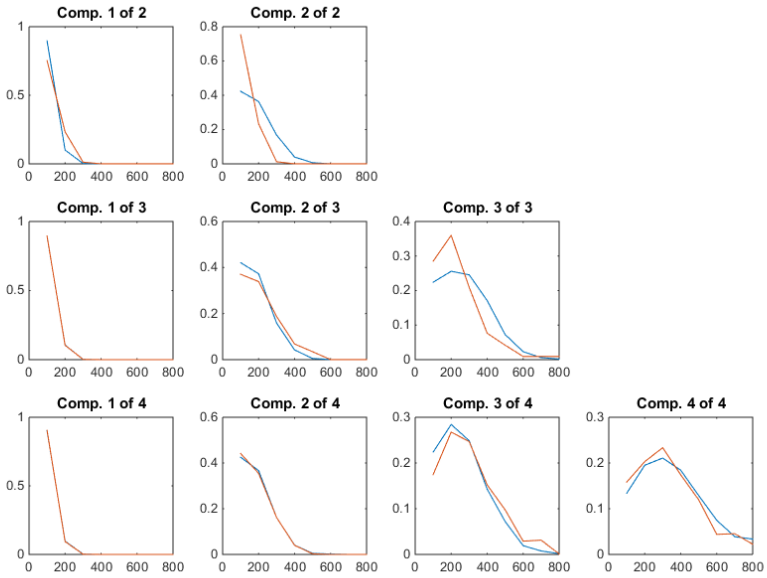

Figure 2: Toy example of component bandwidth pdf recovery using Rayleigh data. The true pdf is shown in blue, the estimated in red.

histogram $\tilde{\mathbf{p}}_y$:

$$\tilde{\mathbf{p}} = \mathbf{F}^H \left(\mathbf{F}\tilde{\mathbf{p}}_y\right)^{1/\bar{N}} \tag{25}$$

where $\bar{N}$ is the median number of path components. Every $\mathbf{p}_j$ is then taken as the first K components of $\tilde{\mathbf{p}}$, limited to the $]0;1[$ range and normalized to sum to unity.

2.6 Path QoS pmf prediction

Given the estimated QoS parameter pmf vectors $\hat{\mathbf{p}}_j$ for each network component j we can estimate the end-to-end QoS pmf $\mathbf{p}_z$ for any candidate path $\mathcal{A}_z$. For additive parameters, this is readily achieved by convolution in the Fourier domain similarly to (7). For bandwidth, we can sample the pdf by evaluating (23) with one observation at each bin b_k, take the exponential and scale by the bin size to recover the probabilities:

$$\mathbf{p}_z(k) = w(k) \times \exp \sum_{j \in \mathcal{A}_z} \log \mathbf{w}_k^T \mathbf{p}_j \times \sum_{j \in \mathcal{A}_i} \frac{\mathbf{a}_k^T \mathbf{p}_j}{\mathbf{w}_k^T \mathbf{p}_j}. \tag{26}$$

3. SKYPE QOS DATASET AND NETWORK MODEL TOPOLOGY

Our dataset contains QoS parameters for approximately 7M Skype video calls. The bandwidth parameter is a passive estimate based primarily on media packet end-to-end one-way delays; it is used in Skype to determine media target encoding rates. Likewise, packet loss, jitter and RTT are measured in-call and end-to-end as well. As such, we are modeling QoS as it is experienced with the Skype application for video calls.

Each call record also includes topology information in terms of *User X Autonomous System (AS)*, *User Y AS*, *Access Technology X*, *Access Technology Y*, and *Relay*. Here, *User X* is the side receiving the media data and recording the QoS metric in question, while *User Y* is the side sending the media data. *Access Technology* is the method by which the user connects to its Internet service provider (ISP) and is assigned to to 1 of 4 different classes: Ethernet, Wifi, Mobile WWAN, or other/unknown. ASs X and Y connect to each other through the Internet "backbone."

Figure 3 shows the network model topology of a call, for bandwidth, jitter, and loss. As up- and downlinks may differ they are modeled independently and therefore the 4 different network types result in 8 network components. With M active ASs in the world, this leads to a maximum of $2M$ network components to model. As of today, M is on the order of 17000 but some of these are only represented sparsely or not at all in the Skype dataset. Therefore, ASs observed less than 10 times are lumped into one "generic" AS group. After doubling due to separate up- and downlink modeling, this results in a total of 22606 components to model. Finally, the end-to-end path may traverse a Skype "relay" that may employ bandwidth limitations and incur other QoS degradations as well. A Skype relay is similar to a TURN server and employed when obtaining a direct peer-to-peer connection is not possible due to

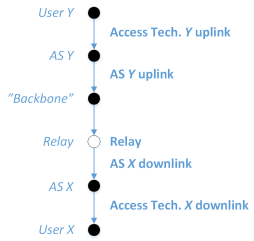

Figure 3: Topology for bandwidth, jitter, and loss metrics. Logical entities are shown in italics on the left with the resulting modeling components in bold-face type on the right.

NAT/Firewall restrictions on both sides of the call. So, a direct peer-to-peer call has 4 network components on its path whereas a relayed has 5. We do not model different relays individually, but all relays as one common component so in total this topology leaves us with 22606 + 8 + 1 = 22615 components to model.

The topology for modeling of RTTs is similar with two exceptions: firstly, RTT is a symmetric measurement, and therefore we do not need to model up- and downlinks separately. Thus there are only 4 network access components and the number of AS components reduces to 11303. Secondly, the topology in Figure 3 does not reflect the geographical distance and thus the propagation delay from X to Y and back. Therefore, for RTT we add another "country bridge" component. With approximately 200 countries in the world there are $200^2/2 = 20000$ such country bridges but again, they are not all well represented in the dataset and we lump those with less than 10 representations into a "generic" group, resulting in 5831 country bridges to model. In total, we have 11303 + 5831 + 4 + 1 = 17139 components to model for RTT.

4. RESULTS

In this section, we report the results of applying the proposed methods to the dataset and topology described above. We use data from 7M Skype video calls, and split the set into roughly 67% for training and 33% for evaluation purposes. For each path in the evaluation (or test) set, we predict both the QoS distribution and the QoS mean. As our measure of accuracy of our prediction of the QoS distribution, we use average log likelihood, while as our measure of accuracy of our prediction of the QoS mean, we use root mean squared error (RMSE). In both cases the average (or mean) is taken over the test set. We explicitly do not evaluate our method in terms of identifying the component distributions, whose ground truth is unknowable in real, large systems.

4.1 Reference baseline

As a reference baseline, we utilize the following: each possible path is enumerated; with the topology of Section 3 one path is e.g. Wifi uplink - AS X uplink - AS Y downlink - Ethernet downlink. With 5 access types at each endpoint, M ASs, and the possibility of relaying, that is $50M^2$ different paths. Now, a reference A pdf is obtained simply from the QoS metrics observed on that exact path in the training set. However, this reference will be accurate only for paths that are well represented in the training set. Therefore, we need a second reference B that is the global distribution of the QoS metric in the training set. The two references A and B are then mixed together into the final reference pdf $C = \lambda A + (1-\lambda)B$, $0 \leq \lambda \leq 1$. λ is bisected to find the optimal value in terms of test set likelihood.

For additive metrics (jitter, loss, RTT) we can moreover use the least squares (LS) estimate of the metric as a second reference. This reference is optimal in the RMSE measure and as such serves as a lower bound to which the proposed method can be benchmarked.

4.2 Results for Skype QoS Dataset

Estimation accuracy results in terms of test set average log-likelihood and RMSE for each of the metrics bandwidth, jitter, packet loss, and RTT are shown in Figure 4. As expected, the proposed method outperforms the baseline when the training set path occurrence count is low whereas the methods perform similarly when the count is high. Also, the proposed method is very close to the optimal LS estimate in terms of RMSE for all additive parameters.

For the packet loss metric the prediction accuracy gain over the baseline is modest. Inspecting the data, we find that packet loss appears to follow an extremely long tailed distribution with the vast majority of calls have zero or very low loss; such a distribution is not easily described as a convolution of individual component distibutions and thus packet loss as available in the dataset may not be suitable for additive modeling. This is supported by the observation that even the LS estimate is only marginally better in terms of RMSE. We also see that with high training set path occurence counts the baseline marginally outperforms the proposed method for the jitter parameter; this is attributed to the fact that jitter is only approximately additive, as described in 2.1.

5. CONCLUSION

The results for the QoS dataset show how the proposed methods greatly reduce the prediction accuracy dependency on training set path occurence count. Thus we conclude that end-to-end QoS can indeed be largely described by contributions from individual network components. This is of great practical importance because by using the methods described herein, it allows OTT

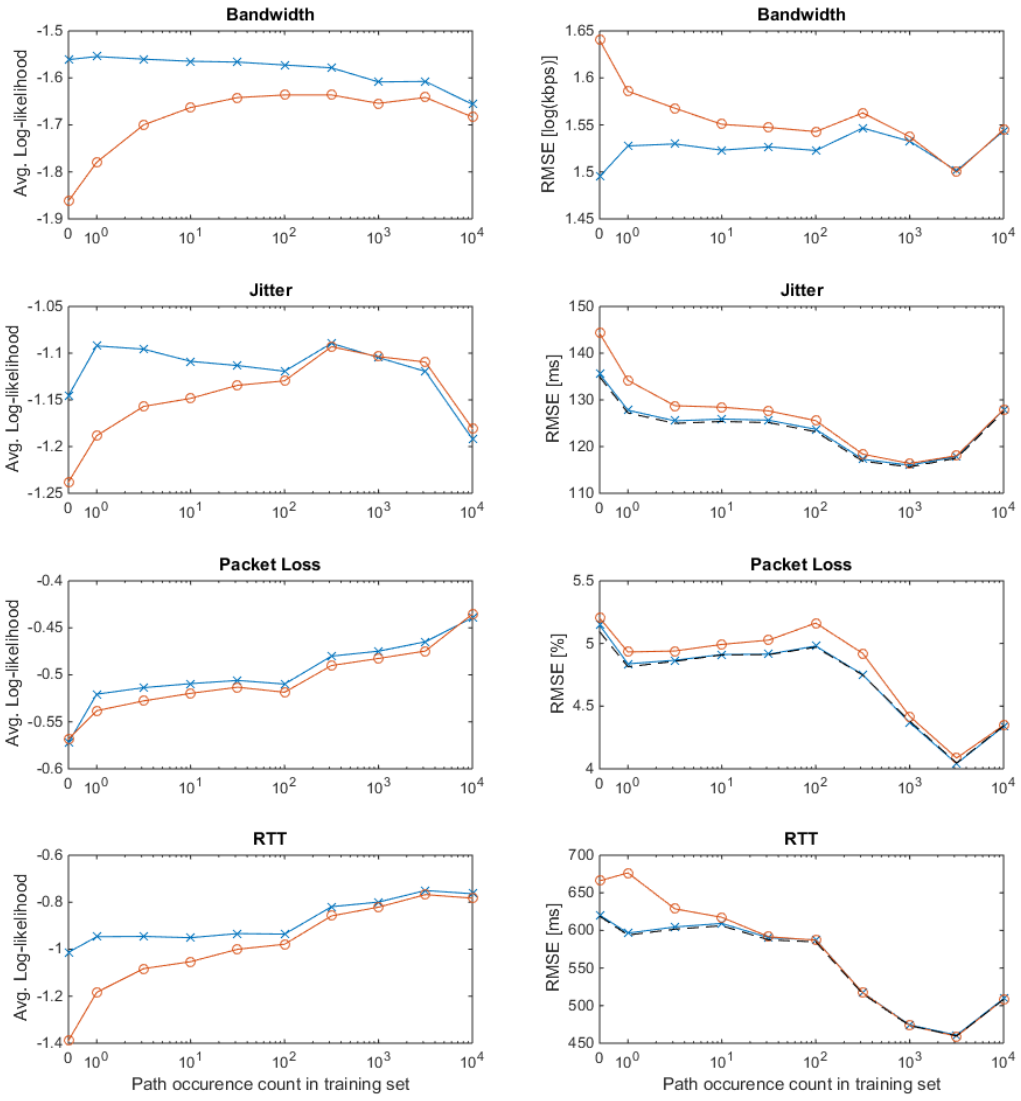

Figure 4: Prediction accuracy in terms of test set log likelihood and RMSE prediction error, proposed method (blue x-marker) vs. baseline (red o-marker) and LS estimate (dashed, only applicable for RMSE and not for bandwidth). Results are shown as a function of the count of path representations in the training set with clustering as indicated by markers.

services to assess the QoS of different route candidates before actually setting up any end-to-end connections.

6. REFERENCES

[1] Elisa Steele. Thanks for making Skype a part of your daily lives – 2 billion minutes a day! Skype blog, http://blogs.skype.com/2013/04/03/thanks-for-making-skype-a-part-of-your-daily-lives-2-billion-minutes-a-day/, April 2013.

[2] TeleGeography. Skype traffic continues to thrive. https://www.telegeography.com/products/-commsupdate/articles/2014/01/15/skype-traffic-continues-to-thrive/, January 2014.

[3] Daniel Genin and Jolene Splett. Where in the internet is congestion? *CoRR*, abs/1307.3696, 2013.

[4] Nate Kushman, Srikanth Kandula, and Dina Katabi. Can you hear me now?!: it must be BGP. *Computer Communication Review*, 37(2):75–84, 2007.

[5] Chris Ritzo. ISP interconnection and its impact on consumer Internet performance. Technical report, Measurement Lab, October 2014.

[6] David G. Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient Overlay Networks. In *18th ACM SOSP*, Banff, Canada, October 2001.

[7] Lakshminarayanan Subramanian, Ion Stoica, Hari Balakrishnan, and Randy Katz. OverQoS: An Overlay Based Architecture for Enhancing Internet QoS. In *1st Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, March 2004.

[8] Krishna P. Gummadi, Harsha V. Madhyastha, Steven D. Gribble, Henry M. Levy, and David Wetherall. Improving the reliability of internet paths with one-hop source routing. In *Proc. OSDI*, pages 183–198, 2004.

[9] Y. Vardi. Network tomography: Estimating source-destination traffic intensities from link data. *J. Am. Stat. Assoc.*, 91:365–377, 1996.

[10] R. Cáceres, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network internal loss characteristics. *IEEE Trans. Information Theory*, 45(7):2462âĂŞ–2480, November 1999.

[11] Francesco Lo Presti, N. G. Duffield, Joe Horowitz, and Don Towsley. Multicast-based inference of network-internal delay distributions. *IEEE/ACM Trans. Networking*, 10(6), December 2002.

[12] N. G. Duffield and Francesco Lo Presti. Network tomography from measured end-to-end delay covariance. *IEEE/ACM Trans. Networking*, 12(6), December 2004.

[13] Mark J. Coates and Robert D. Nowak. Network tomography for internal delay estimation. In *Proc. ICASSP*. IEEE, 2001.

[14] Nick Duffield, Francesco Lo Presti, Vern Paxson, and Don Towsley. Network loss tomography using striped unicast probes. *IEEE/ACM Trans. Networking*, 14(4), August 2006.

[15] Jian Ni and Sekhar Tatikonda. Network tomography based on additive metrics. *IEEE Trans. Information Theory*, 57(12), December 2011.

[16] Mark Coates, Alfred O. Hero III, Robert Nowak, , and Bin Yu. Internet tomography. *IEEE Signal Processing Magazine*, May 2002.

[17] Rui Castro, Mark Coates, Gang Liang, Robert Nowak, and Bin Yu. Network tomography: recent developments. *Statistical Science*, 19(3), 2004.

[18] Venkata N. Padmanabhan, Lili Qiu, and Helen J. Wang. Passive network tomography using Bayesian inference. In *Proc. IMW*. ACM, November 2002.

[19] Tian Bu, Nick Duffield, Francesco Lo Presti, and Don Towsley. Network tomography on general topologies. In *Proc. SIGMETRICS*. ACM, 2002.

[20] Yan Chen, David Bindel, and Randy H. Katz. Tomography-based overlay network monitoring. In *Proc. IMC*. ACM, October 2003.

[21] Aiyou Chen, Jin Cao, and Tian Bu. Network tomography: Identifiability and Fourier domain estimation. In *Proc. Infocom*. IEEE, May 2007.

[22] Gugan Thoppe. Generalized network tomography. In *Proc. Allerton Conference*. UIUC, October 2012.

[23] Ye Xia and David Tse. Inference of link delay in communication networks. *IEEE J. Selected Areas in Communications*, 24(12), December 2006.

[24] Liang Ma, Ting He, Kin K. Leung, Ananthram Swami, and Don Towsley. Identifiability of link metrics based on end-to-end path measurements. In *Proc. IMC*. ACM, October 2013.

[25] Liang Ma, Ting He, Kin K. Leung, Ananthram Swami, and Don Towsley. Monitor placement for maximal identifiability in network tomography. In *Proc. Infocom*. IEEE, April 2014.

[26] Liang Ma, Ting He, Kin K. Leung, Ananthram Swami, and Don Towsley. Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement. *IEEE/ACM Trans. Networking*, 22(4), August 2014.

[27] Liang Ma, Ting He, Ananthram Swami, Don Towsley, Kin K. Leung, and Jessica Lowe. Node failure localization via network tomography. In *Proc. IMC*. ACM, October 2014.

[28] Abishek Gopalan and Srinivasan Ramasubramanian. On identifying additive link metrics using linearly independent cycles and paths. *IEEE/ACM Trans. Networking*, 20(3), June 2012.

[29] Abishek Gopalan and Srinivasan Ramasubramanian. On the maximum number of linearly independent cycles and paths in a network. *IEEE/ACM Trans. Networking*, 22(5), October 2014.

[30] Nick Duffield. Simple network performance tomography. In *Proc. IMC*. ACM, October 2003.

[31] Han Hee Song, Lili Qiu, and Yin Zhang. Netquest: A flexible framework for large-scale network measurement. In *Proc. SIGMETRICS*. ACM, 2006.

[32] Denisa Gabriela Ghiţă. *Practical Network Tomography*. PhD thesis, EPFL, August 2012.

[33] Denisa Ghita, Can Karakus, Katerina Argyraki, and Patrick Thiran. Shifting network tomography toward a practical goal. In *Proc. CoNEXT*. ACM, December 2011.

[34] Denisa Ghita, Katerina Argyraki, and Patrick Thiran. Toward accurate and practical network tomography. In *Proc. LADIS Workshop*, 2012.

[35] Yan Qiao, Guanjue Wang, Xue song Qiu, and Ran Gu. Network loss tomography using link independence. In *Proc. ISCC*. IEEE, 2012.

[36] Mohammad H. Firooz and Sumit Roy. Network tomography via compressed sensing. In *Proc. Globecom*. IEEE, 2010.

[37] Rhys Bowden, Matthew Roughan, and Nigel Bean. Network link tomography and compressive sensing. In *Proc. SIGMETRICS*. ACM, 2011.

[38] Weiyu Xu, Enrique Mallada, and Ao Tang. Compressive sensing over graphs. In *Proc. Infocom*. IEEE, 2011.

[39] Weiyu Xu, Meng Wang, Enrique Mallada, and Ao Tang. Recent results on sparse recovery over graphs. In *Proc. Asilomar*. IEEE, 2011.

[40] M. Amin Khajehnejad, Amir Khojastepour, and Babak Hassibi. Compressed network tomography for probabilistic tree mixture models. In *Proc. Globecom*. IEEE, 2011.

[41] Vidarshana W. Bandara, Anura P. Jayasumana, and Rick Whitner. An adaptive compressive sensing scheme for network tomography based fault localization. In *Proc. ICC*. IEEE, 2014.

[42] Ryoichi Kawahara and Yoshihide Tonomura. Mobile QoS tomography using compressed sensing. In *Proc. Int'l Teletraffic Congress*. ITC, 2014.

[43] Elena Baralis, Andrea Bianco, Tania Cerquitelli, Luca Chiaraviglio, and Marco Mellia. NetCluster: a clustering-based framework for Internet tomography. In *Proc. ICC*. IEEE, 2009.

[44] Kiril Dichev, Fergal Reid, and Alexey Lastovetsky. Efficient and reliable network tomography in heterogeneous networks using bittorrent broadcasts and clustering algorithms. In *Proc. SC*. IEEE, 2012.

[45] Ian H. Dinwoodie. Statistical estimation of internal available bandwidth. *J. Stat. Computation and Simulation*, 78(7):639–652, 2008.

[46] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi. Capprobe: A simple and accurate capacity estimation technique. In *Proc. SIGCOMM*, pages 67–78. ACM, 2004.

[47] M. Jain and C. Dovrolis. End-to-end estimation of the available bandwidth variation range. In *Proc. SIGMETRICS*. ACM, 2005.

[48] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *Proc. SIGCOMM*. ACM, 2000.

[49] X. Zhu and R. Pan. NADA: A unified congestion control scheme for low-latency interactive video. In *Proc. Int'l Packet Video Workshop*. IEEE, December 2013.

[50] Minas Gjoka, Christina Fragouli, Pegah Sattari, and Athina Markopoulou. Loss tomography in general topologies with network coding. In *Proc. Globecom*. IEEE, 2007.

[51] G. Sharma, S. Jaggi, and B. K. Dey. Network tomography via network coding. In *Proc. Information Theory and Applications (ITA)*. IEEE, 2008.

[52] Mohammad H. Firooz, Sumit Roy, Linda Bai, and Christopher Lydick. Link failure monitoring via network coding. In *Proc. Workshop on Network Measurements*. IEEE, 2010.

[53] Hongyi Yao, Sidharth Jaggi, and Minghua Chen. Network coding tomography for network failures. In *Proc. Infocom*. IEEE, 2010.

[54] Jiaqi Gui, Vahid Shah-Mansouri, and Vincent W. S. Wong. Accurate and efficient network tomography through network coding. *IEEE Trans. Vehicular Technology*, 60(6), July 2011.

[55] Pegah Sattari, Athina Markopoulou, and Christina Fragouli. Maximum likelihood estimation for multiple-source loss tomography with network coding. In *Proc. Netcod*. IEEE, 2011.

[56] Jithin R and Bikash Kumar Dey. Exact topology inference for DAGs using network coding. In *Proc. Netcod*. IEEE, 2012.

[57] Hongyi Yao, Sidharth Jaggi, and Minghua Chen. Passive network tomography for erroneous networks: A network coding approach. *IEEE Trans. Information Theory*, 58(9), September 2012.

[58] Pegah Sattari, Athina Markopoulou, Christina Fragouli, and Minas Gjoka. A network coding approach to loss tomography. *IEEE Trans. Information Theory*, 59(3), March 2013.

[59] Peng Qin, Bin Dai, Benxiong Huang, Guan Xu, and Kui Wu. A survey on network tomography with network coding. *IEEE Communication Surveys & Tutorials*, 16(4), 2014.

[60] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[61] Earl Lawrence, George Michailidis, Vijayan N. Nair, and Bowei Xi. Network tomography: a review and recent developments. In Jianqing Fan and Hira L. Koul, editors, *Frontiers in Statistics*. 2006.

[62] Yolanda Tsang, Mark J. Coates, and Robert D. Nowak. Network delay tomography. *IEEE Trans. Signal Processing*, 51(82), August 2003.

[63] Ye Wang, Cheng Huang, Jin Li, Philip A. Chou, and Y. Richard Yang. Qosaas: Quality of service as a service. In *Proceedings of the 11th USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, Hot-ICE'11, pages 6–6, Berkeley, CA, USA, 2011. USENIX Association.

[64] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. Rtp: A transport protocol for real-time applications. STD 64, RFC Editor, July 2003. `http://www.rfc-editor.org/rfc/rfc3550.txt`.

[65] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.