RICE UNIVERSITY

Network Tomography in Theory and Practice

by

Yau-Yau Yolanda Tsang

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE
Doctor of Philosophy

APPROVED, THESIS COMMITTEE:

__________________________
Dr. Richard G. Baraniuk, Chair
Professor of Electrical and Computer
Engineering

__________________________
Dr. Edward W. Knightly
Associate Professor of Electrical and
Computer Engineering

__________________________
Dr. Robert Nowak, Advisor
Associate Professor of Electrical and
Computer Engineering

__________________________
Dr. Rudolf Riedi
Assistant Professor of Statistics

Houston, Texas
April, 2005
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI Microform 3168149

Copyright 2005 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.
Abstract

Network Tomography in Theory and Practice

by

Yau-Yau Yolanda Tsang

Network tomography has recently emerged as a promising method for indirectly inferring network state information from end-to-end measurements. In this thesis, I present novel methodologies for several challenging network inference problems. I also tackle practical problems faced in deploying tomographic techniques in the Internet and provide practical solutions to address and overcome some of these difficulties.

The major contributions are four-fold. First, a passive monitoring technique for estimating internal link-level drop rates is proposed. This approach only requires TCP traces from the end hosts, and it is more effective and less invasive than other tomography schemes. I have demonstrated its effectiveness using ns-2 simulations. I have also conducted theoretical queuing analysis which corroborates the results obtained through simulation experiments.

Second, in delay distribution estimation, a non-parametric wavelet-based approach is developed for estimating link-level queuing delay characteristics. The approach overcomes the bias-variance tradeoff caused by delay quantization, a problem associated with most
existing delay estimation methods. Realistic network simulations are carried out using
ns-2 simulations to demonstrate the accuracy of the estimation procedure.

Third, in order to make tomographic inference techniques more practical, I investigated a Round Trip Time (RTT) based measurement technique. This novel technique does not require clock synchronization and does not require special-purpose cooperation from receivers, enabling deployment of my tomographic tool from any host in the Internet. I demonstrated that my RTT method is effective under a wide range of operating conditions both in an emulation environment and in the Internet.

Finally, to make inference techniques more reliable and robust, I formulated the tomographic data collection process as an optimal experimental design problem, in which a fixed number of network probes are optimally distributed to minimize the squared estimation error of the tomographic reconstruction. Explicit forms for the estimation errors are derived in terms of topology, noise levels, and number and distribution of probes. This analysis reveals the dominant sources causing ill-conditioning and scalability issues in network tomography.
Acknowledgments

I would like to express my deep gratitude to many people. Without their support this thesis would not have been possible. I especially thank my thesis advisor, Robert Nowak for his support, guidance and enlightening discussions throughout my entire graduate studies. He is always there for me not only as an excellent advisor but also a great friend. I thank all my other committee members, Richard Baraniuk, Edward Knightly and Rudolf Riedi who have mentored me during my graduate study and have kindly agreed to be on my thesis committee. I appreciate their support and enlightening comments which led to the enrichment of the thesis.

I appreciate the chance to work with many great people during my graduate study. I am grateful for having the opportunity to work with Mark Coates and Paul Barford, for their constructive advice and numerous insightful discussions. I am grateful for having had the opportunity to work with Nina Taft, Alberto Medina, Richard Gass, Dina Papagiannaki, Sharad Agarwal, Amgad Zeitoun and Christophe Diot during my internship at Sprint ATL. I learned a lot from them and appreciate the friendship thereafter.

My heartfelt thanks go to my parents for their love, support, caring and friendship throughout these years. I know it was hard for them to let their only daughter to study so far away from home, but they did this for my benefit and growth. I thank my fiancee, Jianfeng Liu, who blesses my everyday life with his love, caring and support.

I thank all my friends at Rice and the University of Wisconsin-Madison, especially the
members of the Rice Digital Signal Processing Group members, Rice Networks Group, the
UW-Madison CommDSP Group, and Wisconsin Advanced Internet Laboratory, who have
made my stay so enjoyable. I thank all the staff members at the ECE department and at the
Office of International Students. They have been tremendous help in making my stay at
Rice a breeze. I also want to thank the friendship and hospitality of my friend Sudaryono
Sumargono and his family during my stay in Houston. You are always there when I need
help and support. Finally, I value the company and support of my office-mates, Rui Castro,
Jarvis Haupt, Farbod Khoshnevis, Aline Martin, Mike Rabbat, Becca Willett, Ercan Yildiz
and Ping Yuan. You are the best office-mates that I could ever have dreamt of.
Contents

Abstract ii
Acknowledgments iv
List of Illustrations xi
List of Tables xvi

1 Introduction 1

1.1 Passive Network Loss Tomography ....................... 3
1.2 Nonparametric Network Delay Tomography ............... 5
1.3 Network Radar: Round Trip Time Based Network Tomography ............... 8
1.4 Optimal Network Tomography ................................. 12
1.5 Remarks .................................................. 13

2 Passive Network Loss Tomography 14

2.1 Measurement Framework .................................... 15
2.2 TCP-Based Measurement Framework ....................... 16
2.3 Loss Modeling and Likelihood Analysis ................... 19
2.4 Queuing Analysis of Back-to-Back Losses .................. 21

   Theorem 2.1 ................................................. 22

   Proof of Theorem 2.1 ....................................... 23

2.5 ns-2 Simulation Experiments ............................... 24
2.5.1 Simulation Framework ........................................... 25
2.5.2 Simulation Results ............................................... 28
2.6 Discussion .......................................................... 29
2.7 Related Work ....................................................... 32
2.8 Summary ............................................................ 32

3 Nonparametric Network Delay Tomography ........................................... 33

3.1 Measurement Framework ............................................. 34
3.2 Model Assumptions ................................................... 38
3.3 Measurement Requirements ......................................... 40
3.4 Delay Distribution Inference ........................................ 41
  3.4.1 Likelihood Function ........................................... 43
  3.4.2 MMPLD Density Estimation .................................... 44
  3.4.3 EM Algorithm .................................................. 48
  3.4.4 Fast Fourier Transform based EM Algorithm ............... 51
3.5 Simulation Experiments ............................................. 54
3.6 Related Work ....................................................... 58
3.7 Summary ............................................................ 62

4 Network Radar: Round Trip Time based Network Tomography ................. 64

4.1 Measurement Framework ............................................. 64
4.1.1 Challenges ................................................. 65
4.1.2 Tomography Basics ........................................ 67
4.1.3 Round Trip Time Tomography ............................ 69
4.1.4 Measurement Methodology ............................... 71

4.2 Proposed Estimator and Performance Analysis ............. 72

Proposition 4.1 ............................................... 72
Proof of Proposition 4.1 ....................................... 73
Definition 4.2 .............................................. 74
Proposition 4.3 ............................................... 74
Proof of Proposition 4.3 ....................................... 74
Proposition 4.4 ............................................... 76
Proof of Proposition 4.4 ....................................... 76
Proposition 4.5 ............................................... 77
Proof of Proposition 4.5 ....................................... 78

4.3 Experimental Evaluation .................................... 79

4.3.1 Timestamping Mechanism ................................ 80
4.3.2 Emulation Experiment .................................... 81
Data Processing ............................................... 82
Results ...................................................... 86
Operating Condition Study .................................... 87
4.3.3 Internet Experiments .................................... 95
5 Optimal Network Tomography

5.1 Measurement Infrastructure

5.2 Problem Formulation

5.3 Optimal Experimental Designs

5.3.1 D-Optimality

5.3.2 A-Optimality

5.3.3 Constrained Optimization

5.3.4 Generalized Lagrange Multiplier

Theorem 5.5
5.4 Scalability Study ........................................ 128
  Lemma 5.6 .................................................. 128
  Proof of Lemma 5.6 ....................................... 128
  Lemma 5.7 .................................................. 129
  Proof of Lemma 5.7 ....................................... 129
5.4.1 Heuristics for Computing Unknown Error .......... 131
5.5 Simulation Experiments .................................. 131
  5.5.1 Simulation Framework ................................ 132
  5.5.2 Simulation Results ................................... 132
5.6 Related Work ........................................... 133
5.7 Summary ................................................ 135

6 Conclusion ................................................ 136
Illustrations

1.1 An example network topology with a single source (node 0), 4 internal routers, and 7 receivers. .................................................. 2

1.2 (a) ns-2 delay measurements using 170 packets on link 9 for network depicted in Fig. 3.1. Horizontal axis is (discretized) delay time and vertical axis denotes the number of occurrences of a particular delay measurement. (b) Discretized pmf with 16 equal-width bins. (c) Discretized pmf with 64 bins. (d) Nonparametric density estimate proposed in this paper obtained by direct estimation using link delays. ........................................ 7

2.1 Simulation Results. True and estimated link-level success rates of TCP flows from source to receivers for all traffic scenarios. In each subfigure, the three panels display for each link 1 – 11 (horizontal axis): (top) true and estimated success rates using drop-tail queues, (middle) true and estimated success rates using RED queuing policy, and (bottom) mean absolute error for each link (RED and drop-tail). ................................................................. 27

2.2 The performance error (mean absolute error averaged over all links) versus measurement period. ......................................................... 28

2.3 An example of “grouping” receivers to reduce network complexity. ........ 30
3.1 Tree-structured network topology used for ns-2 simulation experiments. Source (node 0) transmits to 6 receivers (nodes 6 – 11). Link speeds in Mb/s are shown next to the links. Link i connects node i to its parent node, e.g. link 9 connects nodes 5 and 9.

3.2 (a) End-to-end delay histogram (packets sent from Rice University to Michigan State University). (b) Difference between delays of the two packets in packet pairs. Measurements were made using the netdyn tool.

3.3 The factor graph used in the message-passing algorithm for a measurement made by a packet pair sent to nodes 6 and 7 in the network of Fig. 3.1. Measurements are available at nodes 6 and 7; the nodes $p_i^{(r)}$ contain current pmf estimates; and node $c_{a,b}$ indicates the convolutional relationship between nodes $d_a$, $d_b$ and $z_b$.

3.4 Comparison between true pmfs (solid) and estimated pmfs (dashed). Top panel shows true pmf and MMPL (calculated using 512 bins); bottom panel shows true pmf and MLE (calculated using 16 bins). 16 bins is determined as the bin size at which the MLE obtains the best fit. (a) Link 5. (b) Link 7. (c) Link 9.

3.5 $L_1$ error criterion averaged over 25 simulations (means and standard deviation) for link 5, 7 and 9. Solid line is MMPL, dashed line is MLE (16 bins), dotted line MLE (64 bins).

3.6 A larger tree-structured network topology used for ns-2 simulation experiments. Source (node 0) transmits to 20 receivers (nodes 19-38). Link speeds in Mb/s are shown next to the links.
3.7 Comparison between true pmfs (solid) and estimated pmfs (dashed). Top panel shows true pmf and MMLE (calculated using 512 bins); middle panel shows true pmf and MLE (calculated using 64 bins); bottom panel shows true pmf and MLE (calculated using 512 bins). 512 bins is determined as the bin size at which the MLE obtains the best fit. (a) Link 1. (b) Link 20. (c) Link 31. 59

3.8 $L_1$ error criterion averaged over 20 simulations (means and standard deviation) for some terminating links. Solid line is MMLE, dash-dot line is MLE (512 bins), and dotted line is MLE (256 bins). 60

3.9 The cdf estimates obtained from direct measurement (solid) to the tomographic one (dotted). (a) Link 1. (b) Link 20. (c) Link 31. 61

4.1 One way tomographic delay variance estimation in a standard one sender (0) two-receiver (1, 2) network. Variances on shared ($\sigma^2_{\text{one-way},a}$) and unshared segments ($\sigma^2_{\text{one-way},1}$, $\sigma^2_{\text{one-way},2}$) are noted. 68

4.2 Round trip time tomographic delay variance estimation in a standard one sender (0) two-receiver (1, 2) network. Variances on shared ($\sigma^2_{\text{RTT},a}$) and unshared segments ($\sigma^2_{\text{RTT},1}$, $\sigma^2_{\text{RTT},2}$) are noted. 69
4.3 The laboratory network configuration includes 5 routers and 9 PCs. The sending host is 0 and the receivers are 1 and 2 (logical topology in gray). The boxes $xT$ denote cross-traffic generators and the balls $R$ denote CISCO 7200/7500/12000 series routers. $S1$ and $S2$ denote measurement systems used to validate the performance of the RTT based tool and DAG denotes the DAG measurement system placement.

4.4 Data processing flowchart.

4.5 Plot of standard deviation from direct measurement (horizontal axis) vs. estimated delay standard deviation using RTT-tomography tool (vertical axis).

4.6 Verification of timestamping mechanism using $\texttt{tcpdump}$ utility and hardware-base reference systems using DAG cards.

4.7 Significant of background traffic in the estimates. High SNR symbolizes high accuracy in the estimates. The signal to noise ratio (SNR) increases when the background traffic (correlation) on the shared segment increases. Unshared segments had moderate load in these experiments.

4.8 Plot of directly measured delay standard deviation (vertical axis) and the estimated one (horizontal axis) on the shared link when the spacing between the packets are $3\mu s$ (circle) and $100$ ms (triangle) apart at the sender.

4.9 Plot of delay standard deviation on the shared link measured from sender to the branching node destined to receiver 1 against that to receiver 2.
4.10 Plot of signal to noise ratio (vertical axis) given different number of packet pair probe measurements. .................................................. 94

4.11 An example of topology identification illustration with 3 receiver nodes. ........ 99

4.12 Comparison between (a) “true” topology and (b) estimated topology for 6 universities on the Abilene Network. The university homepages indexed from 1 – 6 are Oklahoma University (OU), Virginia Commonwealth University (VCU), Vermont University (UVM), Utah State University (USU), University of Washington (UWash) and University of Idaho (UIdaho). Probes are sent from Rice University. .................................................. 100

5.1 An example of tree-structured topology with single sender (node 0) and 33 receivers (node 6 through 18). The logical topology is composed of only the solid nodes. The edge links (thinner lines) are slower and the internal links (thicker lines) are faster. .................................................. 112

5.2 Comparison of the squared error given a probe budget with uniform and optimal probing schemes). .................................................. 133
Tables

4.1 An example of the lab-based experiments. The average RTT to node $i = 1, 2$ is $\bar{y}_i$. The standard deviation of RTT to node $i = 1, 2$ is std$(y_i)$.

The SNR is given as the ratio between the estimated shared link delay variance $\hat{\sigma}_2^2$ and its confidence interval $\sqrt{\hat{\delta}_2^2}$.

4.2 Average SYN-ACK generation delay and its variance with varying end host load.

4.3 An example of the Internet experiments from node UW. The packet pairs are destined to Receiver 1 and 2. Their mean round trip times are $\bar{y}_1$ and $\bar{y}_2$ and their variances are $\sqrt{\hat{\sigma}^2_1}$ and $\sqrt{\hat{\sigma}^2_2}$. The estimated shared link variance is shown in $\hat{\sigma}^2_2$ and its confidence (std) in $\sqrt{\hat{\delta}^2_2}$. The SNR is shown in the last column.

4.4 An example of the Internet experiments from node Rice. The packet pairs are destined to Receiver 1 and 2. Their mean round trip times are $\bar{y}_1$ and $\bar{y}_2$ and their variances are $\sqrt{\hat{\sigma}^2_1}$ and $\sqrt{\hat{\sigma}^2_2}$. The estimated shared link variance is shown in $\hat{\sigma}^2_2$ and its confidence (std) in $\sqrt{\hat{\delta}^2_2}$. The SNR is shown in the last column.
Chapter 1

Introduction

Accurately characterizing the performance of a large-scale network is essential if it is to be successfully managed and controlled. The characterization must extend further than path-level behavior; it is necessary to acquire information about internal (link-level) performance. Spatially localized information about network performance, such as link loss rates, queuing delays and available bandwidths, plays an important role in isolating network congestion and detecting performance degradation. Routing algorithms, servicing strategies, security procedures and performance verification can benefit from monitoring techniques that report such information. One way to achieve this is to gather statistics at as many internal routers as possible, but the collection and compilation of such statistics is an onerous and expensive task. Often it proves impossible for any one organization or individual to collect relevant statistics when various parts of a network are administered by different parties.

An attractive alternative is to infer the internal performance from end-to-end measurements that are comparatively easy to make. This has prompted several groups to investigate methods for inferring internal network behavior (1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17). This problem is often referred to as network tomography; see (5) for an overview of work in this area.
In this thesis we focus on unicast network tomography. Earlier inference methodologies focused on multicast routing. In multicast routing, packets are delivered from sender to the receivers in one send operation. Along the path, probe packets are duplicated as needed as the paths diverge (10; 17). Although multicast methods show promise for network performance inference, these techniques are often impractical in real networks. Many routers do not support multicast traffic and, if they do, they treat the packets differently from the majority of the traffic which is based on unicast routing. Therefore, inferences drawn from multicast routing may poorly reflect the actual network performance as observed by most traffic. However, the use of single unicast packets does not provide correlated measurements as do multicast packets. This motivates the use of back-to-back (closely time-spaced) unicast packets, which mimic the behavior of multicast packets to some degree.

![Diagram](image)

Figure 1.1: An example network topology with a single source (node 0), 4 internal routers, and 7 receivers.

If one were to trace the paths of these probe packets from sender to receiver, they would form a tree with the root at the sender, a common trunk and the leaves at the receivers.
Figure 1.1 depicts an example of this form of topology; the network appears to the source as a tree. The nodes of the tree correspond to the source (node 0), internal routers (nodes 1–4) and receivers (nodes 5–11). We define a “link” as the connection between any two adjacent nodes in the tree, deem the set of links connecting a source and any receiver a “path”, and a subset of connected links in a path is referred to as a “subpath”. The tree of Figure 1.1 does not necessarily depict all routers encountered by packets traveling from the source to receivers. It is possible that a number of routers are passed as a packet travels from node 1 to node 3, for example.

In this thesis, we provide novel methodologies for several challenging network inference problems including link-level loss rate and delay distribution estimation. We also address some practical problems faced in deploying tomographic techniques in the Internet, and provide practical solutions to address and overcome some of these difficulties.

1.1 Passive Network Loss Tomography

In link loss rate estimation, we propose a passive inference scheme, a non-intrusive method that do not require packets to be injected into the network. In this work, we extend the procedure proposed in (7). Although the unicast network tomography proposed in (7) suggested that the collection of statistics is possible using both active probing and passive sampling of existing traffic, the experiments and performances in reported in previous work use active probing strategies (7; 10; 18). Moreover, a number of issues encountered in performing passive sampling have not been not addressed. In Chapter 2, we concentrate
on the inference of internal link losses from passive unicast end-to-end measurement. The motivation for passive inference is two-fold. Firstly, there is a risk that for a substantial network the insertion of the large number of probes required for accurate estimates might significantly impact the performance of the network (perturbing the very quantity to be estimated). Secondly, the sampling of existing flows potentially offers an opportunity to estimate the losses experienced by the existing flows (which can be substantially different from those experienced by inserted probes). We propose a new, truly passive methodology for unicast network tomography, and we assess the feasibility and performance our approach through theoretical analyses and ns-2 simulation experiments.

We also make an important modification to the framework proposed in (7). The success of the framework, as discussed in Chapter 2, hinges on a conjecture. The conjecture states that the (causal) conditional probability of the second (in temporal order) packet in a pair successfully traversing a subpath given that the first packet did so is higher than the unconditional success probability (and close to one). In this work, we consider the anti-causal conditional probability instead (the probability that the first packet is successful given that the second is). We are able to prove a critical result. The anti-causal conditional success probability is always greater than the unconditional success probability, irrespective of the nature of the traffic arrival process and queueing service strategy (assuming a drop-tail queue). If a single link (direct connection between two neighboring routers) is considered and the packets are truly back-to-back, then the conditional probability is one. The conditional probability then decays according to the nature of the traffic, eventually reaching the
unconditional probability.

We use extensive ns-2 (19) simulations to explore various aspects of our new methodology’s performance. We assess whether sufficient back-to-back packet statistics can be extracted from existing TCP flows between the source and receivers. We quantify the performance of the inference algorithm in terms of mean absolute error.

1.2 Nonparametric Network Delay Tomography

Queuing delays are one of the most critical performance characteristics. Optimizing communication network routing and service strategies requires knowledge of the queuing delay at different points in the network. Measuring end-to-end (source to receiver) delays using timestamps (14; 20; 21) is relatively easy and inexpensive in comparison to internal measurements, although there are, of course, measurement issues that must be addressed.

In Chapter 3, we introduce a new methodology for network tomography, specifically, estimating the probability distribution of the queuing delay on each link based on end-to-end unicast packet pair measurements. Our approach employs unicast, end-to-end measurement of back-to-back packets, possibly destined for different receivers, but sharing a common set of links in their paths. The two packets should experience approximately the same on each shared link in their path.

In this chapter we describe a nonparametric framework for the inference of internal delay distributions based on unicast end-to-end measurement. By nonparametric we mean that the number of parameters or the degrees of freedom diverges as a function of the num-
ber of delay measurements (22). Most work to date in network tomography is based on parametric models. Parametric models assume that the measured traffic data depends on a finite number of parameters. For example, earlier work in delay distribution estimation has been based on discretized (or quantized) delay measurements, with internal delay distributions modeled as discrete probability mass functions (pmfs) (8; 9; 16). In this context, the parameters are simply the probabilities associated with each pmf. It has been our experience, as well as that of others (23; 24), that no sufficiently simple parametric model is capable of portraying the wide variety of internal delay distributions observed in practice, thus motivating the consideration of nonparametric or continuous models. The complex nature of network delay distributions is evident in the simulated network measurements and estimates depicted in Figure 1.2.

Our methodology offers several significant advantages over existing methods: (1) it utilizes unicast measurement so that inferred performance reflects the experience of the majority of network traffic; (2) the estimation procedure is nonparametric and very flexible, in that it is capable of recovering densities from a broad range of function spaces including Bounded Variation (BV) functions and Besov spaces, which include both smooth and piecewise smooth densities; (3) the use of a Multiscale Maximum Penalized Likelihood Estimator (MMPLE) provides a computationally fast method for balancing the bias-variance trade-off and has been shown to be nearly optimal for density estimation in the above-mentioned function spaces (25; 26); (4) we develop a new fast Fourier transform based implementation of the Expectation-Maximization (EM) algorithm for the network
Figure 1.2: (a) ns2 delay measurements using 170 packets on link 9 for network depicted in Fig. 3.1. Horizontal axis is (discretized) delay time and vertical axis denotes the number of occurrences of a particular delay measurement. (b) Discretized pmf with 16 equal-width bins. (c) Discretized pmf with 64 bins. (d) Nonparametric density estimate proposed in this paper obtained by direct estimation using link delays.
tomography problem which, in combination with MMPL, leads to a worst-case overall complexity of $O(MN^2 \log N)$ where $M$ is the number of links in the network and $N$ is the number of packet pair measurements. In general, the complexity is substantially less than this (see Section 3.4.4 for clarification). We demonstrate the flexibility and accuracy of the nonparametric approach through ns-2 (19) simulation.

1.3 Network Radar: Round Trip Time Based Network Tomography

Researchers interested in measuring the behavioral and structural characteristics of the Internet are faced with many challenges, not the least of which is an inherent lack of open instrumentation. Similarly, network operators charged with the responsibility of identifying and diagnosing problems in their networks have an on-going need for tools that make this process faster and more efficient. These issues motivate the development of novel methods and systems for measuring and monitoring network characteristics beyond the confines of a single network.

These tools report network characteristics based on either passive measurements of traffic at a given vantage point, or by measuring the response to probe packets emitted by the tool itself. Tomographic inference techniques are shown to be useful in deducing link-specific information such as packet loss rates, packet delay characteristics and network topology. In Chapter 4, we describe a new idea of using round trip times for network tomography. In this work we develop the analytic foundation for round trip time network tomography, address the various practical considerations in its use, build a tool for Round
Trip Time (RTT) tomography and test it both in a lab environment and in the Internet. RTT-based tomography sends back-to-back packets to different pairs of receivers, collects response packets at the sender and measures round trip times of the probes. We call this approach Network Radar since it is analogous to standard radar. Like prior tomographic tools, the objective is to infer characteristics on the outward shared path between sender and receivers. The important benefit of this approach is that under appropriate operating conditions, it eliminates the need for special-purpose cooperation between senders and receivers. This would allow RTT-based tomographic tools to not only be more widely available to users, but would also greatly expand the number of paths over which tomography could be used. In this work, we assess the validity and capabilities of Network Radar through delay variance estimation. There are other link performance parameters which might be of interest. However, delay variance estimation is the simplest. Extensions of the technique to other performance characteristics, such as link loss rates will be briefly mentioned. The goal of the work is to investigate a technique which can be used widely in the Internet under real and practical conditions. We develop a new analytic framework that automates the tomographic inference process. This framework includes the capability to estimate the confidence of individual delay estimates. The practical impact of this is that it enables inaccurate delay estimates, that would otherwise skew results, to be automatically detected and discarded. As a natural extension and a proof of performance using our tool, it is shown that we can identify the underlying topology by combining sets of measurements.

To conduct our evaluation of RTT tomography, we develop a packet-pair measurement
tool which can be widely used in the Internet. Our tool collects round trip time measurements by using the Transport Control Protocol (TCP) connection setup mechanism. The tool initiates a connection by sending a back-to-back pair of synchronize (SYN) packets to the HTTP port on two different receivers. It then measures the round trip times to those hosts based on receipt of the corresponding synchronize-acknowledge (SYN-ACK) packets. The tool then immediately closes each connection by sending reset (RST) packets. The benefits of the TCP connection setup mechanism are that, 1) HTTP is a service that is widely available in the Internet, 2) TCP packets destined for the HTTP service are rarely blocked by Internet Service Providers (in contrast to ICMP packets), and 3) SYN-ACK packets are typically generated with very little delay by receivers (27). The tool sends a series of packet pairs to the receivers at fixed intervals. Both SYN and SYN-ACK packets are timestamped using a packet filter and RTTs are extracted by simple time differencing. While packet filters and their associated timestamps are known to be problematic (28), careful clock disciplining can bring the accuracy within acceptable limits. Using this tool, we perform extensive studies on the performance of our estimator in both emulation environment and in the Internet.

The experiments were conducted on a network of Cisco routers and PC hosts configured in a simple tree topology. The experiments revealed that Network Radar is effective over a wide range of traffic conditions. Most importantly, using our confidence estimator, we were able establish a correlation between the delay estimator and the number of packet pair measurements. Specifically, as the number of packet pair measurements is increased by a
factor of 10, the standard deviation of the delay estimator goes down by a factor of \( \sqrt{10} \) even under highly variable traffic conditions.

Our experiments in the Internet examine the application of RTT tomography to the problem of logical topology discovery (29). In this case, we use a clustering algorithm based on measured shared segment delays to establish logical connectivity between nodes. We choose this application since it gives us an opportunity to validate our results using other tools - delay or loss tomography would have required instrumentation that was not available in the wide area. Our experiments consist of randomly selecting a set of destinations and then using Network Radar to discern topological connectivity between a source and those destinations. We validated the results using traceroute. The results show that Network Radar is very effective at establishing logical topology.

In summary, in this work, we make the following contributions. First, we provide an analytic framework for a tomographic inference method based on round trip time measurements. This framework includes a confidence estimator that enables measurements that would otherwise skew results to be discarded. Second, we describe a tool, Network Radar, for conducting RTT tomography experiments that can be deployed widely on end hosts. Third, we investigate the robustness of Network Radar and show that it is effective over a wide range of operating conditions. The primary implication of this work is that tomography based on round trip time measurements should now be used widely for both research and operational purposes.
1.4 Optimal Network Tomography

All techniques in unicast network tomography explore the correlation in packets and devise algorithms in order to obtain accurate estimates with low complexity. The scalability of such approaches, however, may ultimately limit their utility. Most network tomography schemes considered to date explicitly or implicitly distribute the probes uniformly among the subpaths. We give an optimal strategy for probe allocation based on minimizing the squared error of the network tomography estimate. We show that this can produce significantly better results than uniform probe distribution. Moreover, our analysis of the probe allocation problem also sheds light on the scalability of network tomography. We show that the number of probes required to achieve a given level of accuracy grows linearly with the number of receivers involved in the experiment.

Network tomography schemes considered to date actively distribute the probes uniformly or randomly among the subpaths or passively sample data in existing connections. This work poses the network tomography problem as the optimal probe allocation problem. In Chapter 5, we describe an optimal strategy for probe allocation given a budget of $N$ measurements for the network tomography estimation problem based on unicast end-to-end measurements. Our scheme is optimal in the sense that for a given $N$ measurements, the squared error of the network tomography estimation is minimized.

Specifically, the goal is to minimize the squared error of the desired link estimates by optimally allocating active probes. The main contributions of this work are: (1) we formu-
late network tomography as an optimal allocation problem; (2) we derive explicit forms for
squared error of link estimation errors in terms of topology, noise levels, and number and
distribution of probes; (3) we investigate dominant sources of ill-conditioning and scalabil-
ity issues of network tomography; (4) we apply resource allocation optimization techniques
to design optimal probing schemes; (5) we develop algorithms for optimal probe allocation
based on topological considerations and noise characteristics; (6) ns-2 (19) simulation
demonstrates the potential gains achievable through optimized probe allocation.

1.5 Remarks

Each chapter (Ch. 1 – 4) is treated independently and ends with a discussion of related
works and a summary of our contributions to the corresponding field. In Chapter 5, we
conclude and provide some insights about future research directions. The reader should
also bear in mind that the notations in each chapter are treated independently while pursuing
the thesis. The notations are clearly defined in each chapter and there is no reuse of symbol
within each chapter. However, we cannot completely avoid using the same symbols for
different quantities in these chapters.
Chapter 2

Passive Network Loss Tomography

In this chapter, we provide an overview of the unicast loss inference problem, describe the back-to-back measurement framework proposed in (7), and review the loss modeling framework. We illustrate the problem of unicast loss inference by considering the simple case in which a single source sends packets to multiple receivers. The problem and methodology are readily extended to the multiple-source case.

This chapter is organized as follows. In Section 2.1, we review the basic unicast network tomography problem and the technical issues involved. In Section 2.2, we describe the new passive measurement framework, focusing on the collection of statistics from TCP flows. In Section 2.3, we apply the loss modeling and likelihood analysis techniques devised in (7) to our new approach. In Section 2.4, we perform a queuing analysis of the back-to-back measurement procedure, concentrating on the anti-causal conditioning proposal. In Section 2.5, we describe and discuss the results of ns-2 simulations exploring the performance of the passive measurement and inference methodology. In Section 2.6, we discuss some of the limitations of the passive scheme, and propose some potential remedies. In Section 2.7, we describe the related works. Conclusions are made in Section 2.8.
2.1 Measurement Framework

We consider the situation where measurements can only be made at the edge of the network and assume that the routing (and thus the topology) table is fixed for the duration of the measurement. The goal of passive unicast loss inference is to estimate the loss rates on the internal links of the network using solely passive sampling of the existing traffic. Estimating the loss rates is equivalent to estimating success rates, and henceforth we shall speak solely of success rates, since they simplify mathematical expressions in the proposed framework. (The success rate is simply one minus the loss rate).

It is straightforward to estimate path success rates, but, unfortunately, there is no unique mapping of the path success rates to the success rates on individual links in the path. To overcome this difficulty, the authors in (7) propose a methodology based on measurements of back-to-back packet pairs. These measurements provide an opportunity to collect more informative statistics that can help to resolve the links.

Back-to-back packet pairs have been utilized for inferring a number of network performance metrics (30; 31; 32; 33). A back-to-back packet pair refers to two closely time-spaced packets, possibly destined for different receivers, but sharing a common set of links in their paths. If two back-to-back packets are sent across a link and the one of the pair was received, then it is highly likely that the other packet was also received. Moreover, we expect that the conditional success probability of one packet (given that the other was received) may often be close to one. This observation has been verified experimentally in
real networks (33). Results can also be established theoretically without assumptions about traffic arrival and queuing service processes (see Section 2.4). A unicast network tomography method that exploits the correlation between back-to-back packet losses is developed in (7). The basic idea is quite straightforward. Suppose two back-to-back packets are sent to two different receivers. The paths to these receivers share a common set of links from the source but later diverge. If one of the packets is dropped and the other successfully received, then (assuming strong correlation of losses on common links) one can infer that the packet was probably dropped on one of the unshared links. This enables the resolution of losses on individual links.

2.2 TCP-Based Measurement Framework

Earlier unicast network tomography schemes have focused on active probing (7; 10; 18) (insertion of additional packets whose sole purpose is for measurement), although the possibility of passive measurement (collecting measurements directly from existing traffic) was mentioned, but not investigated, in (7). Although active probing allows one to control the timing and nature of the measurements, thus assuring that sufficient measurements are made, it imposes an extra burden on the network which may make it impractical in many situations. Furthermore, active probing will disrupt the transmission of normal traffic, which may lead to biased estimates of the true losses (experienced in the absence of probe packets). Also, probe traffic may experience significantly different losses than normal traffic (e.g., TCP) since the temporal structure of the probing is specified by the experimenter,
and is generally different than that of normal traffic. For example, TCP often results in clusters of packets separated by a significant (round-trip) time, contrary to a uniform or Poisson probing scheme.

We propose a passive traffic monitoring/sampling scheme in order to circumvent the problematic issues surrounding active probing. We focus on TCP-based measurement, because we are interested in estimating the link-level losses experienced by TCP connections flowing from the source to the receivers. When estimating TCP losses, the spacing of measurements should clearly not be uniform or exponentially distributed. Where the TCP traffic to a particular connection is dense, we should make many measurements; where light, only a few. The measurement process should ideally be a subsampled version of the true traffic. This is impossible to achieve exactly, because we have other constraints. For example, we need to make back-to-back measurements and we need to ensure that measurements are sufficiently spaced to provide the approximate inter-pair temporal independence assumed in our statistical model.

The situation we consider is one where the source has numerous contemporaneous TCP connections with a number of receivers. We want to extract as many informative measurements as we can from the existing TCP traffic. We concentrate first on extracting the important packet-pair measurements, which are less common than the isolated packet measurements. We first inspect the sending times of the TCP traffic at the source and decide that two packets form a packet-pair if their time-spacing is less than a threshold $\delta_I$ seconds. This threshold is dependent on the sending rate of the source. Commencing at the start
of the measurement period, we sweep forward in time, seeking and identifying the first packet-pair. We then step forward by a fixed time interval $\Delta_t \gg \delta_t$ and begin to search for the next pair. In this way, we ensure that the pairs we include in our analysis are separated by a reasonable time interval, making the assumption of statistical independence between pairs more realistic.

Following the collection of these pairs, we include any isolated packets that do not violate the time separation requirement. In general, we observe the number of such packets to be significantly larger than the number of pairs. We are now in a position to define the following statistics. For the purpose of anti-causal conditioning (see Section 2.4), let $n_{i,j}$ denote the number of pairs wherein the first packet is sent to receiver $i$ and the second to receiver $j$ AND the second packet is successfully received. Let $m_{i,j}$ denote the number of these pairs in which both packets are successful. Let $n_i$ denote the number of isolated packets sent to receiver $i$ and let $m_i$ denote the number successfully received. Collecting all the measurements, define:

$$\mathcal{M} \equiv \{m_i\} \cup \{m_{i,j}\} \quad \text{and} \quad \mathcal{N} \equiv \{n_i\} \cup \{n_{i,j}\},$$

(2.1)

where the index $i$ alone runs over all receivers and the indices $i, j$ run over all pairwise combinations of receivers in the network.
2.3 Loss Modeling and Likelihood Analysis

The modeling frameworks of (7; 10; 17; 18) assume a simple Bernoulli loss model for each link for individual packet transmissions. The unconditional success probability of link \(i\) (the link into node \(i\)) is defined as

\[
\alpha_i \equiv \text{Pr(} \text{packet successfully transmitted from } \rho(i) \text{ to } i),
\]

where \(\rho(i)\) denotes the index of the parent node of node \(i\) (the node above \(i\)-th node in the tree; e.g., referring to Figure 1.1, \(\rho(1) = 0\)). A packet is successfully sent from \(\rho(i)\) to \(i\) with probability \(\alpha_i\) and is dropped with probability \(1 - \alpha_i\). Loss processes on separate links are modeled as mutually independent.

If a back-to-back packet pair is sent from node \(\rho(i)\) to node \(i\), then we define two conditional success probabilities:

\[
\beta_i \equiv \text{Pr(} 2\text{nd packet } \rho(i) \rightarrow i \mid 1\text{st packet } \rho(i) \rightarrow i),
\]

\[
\gamma_i \equiv \text{Pr(} 1\text{st packet } \rho(i) \rightarrow i \mid 2\text{nd packet } \rho(i) \rightarrow i),
\]

where 1st and 2nd refer to the temporal order of the two packets, and \(\rho(i) \rightarrow i\) is shorthand notation denoting the successful transmission of a packet from \(\rho(i)\) to \(i\). In earlier work (7; 18), only the first (causal) conditional success probability was considered. Here we draw an important distinction between causal \(\beta_i\) and anti-causal \(\gamma_i\) conditional probabilities. We
demonstrate in Section 2.4 that the anti-causal conditioning offers a significant advantage, and focus on the anti-causal case in the remainder of the chapter.

The sampled TCP flows and corresponding traffic statistics lead to the following likelihood function. We denote the collections of the unconditional and (anti-causal) conditional link success probabilities as $\alpha$ and $\gamma$, respectively. The joint likelihood of all measurements is given by

$$l(\mathcal{M} | \mathcal{N}, \alpha, \gamma) = \prod_i Bi(m_i | n_i, p_i(\alpha)) \times \prod_{i,j} Bi(m_{i,j} | n_{i,j}, p_{i,j}(\alpha, \gamma)),$$

where $Bi(m | n, p) \equiv \binom{n}{m} p^m (1-p)^{n-m}$, the binomial likelihood function, $p_i(\alpha)$ is a product of the unconditional success probabilities in the path from the source to receiver $i$, and $p_{i,j}(\alpha, \gamma)$ is a product of conditional success probabilities (on the common links in the paths to receivers $i$ and $j$) and unconditional success probabilities on the links to $j$ not shared in the path to $i$.

The EM Algorithm developed in (7) can be used to compute maximum likelihood estimates of $\alpha$ and $\gamma$. Beginning with an initial guess for $\alpha$ and $\gamma$, the algorithm is iterative and alternates between two steps until convergence. The Expectation (E) Step computes the conditional expected value of the unobserved packet losses at internal nodes given the observed data, under the probability law induced by the current estimates of $\alpha$ and $\gamma$. The Maximization (M) Step combines the observed (path) losses and expected unobserved (internal) losses to compute new estimates of $\alpha$ and $\gamma$. Each iteration of the EM Algorithm
is close to $O(N)$ in complexity, where $N$ is the number of nodes in the network. The exact complexity depends on the topology of the network. For example, the complexity is $O(N \log^2 N)$ for a binary tree network topology. Our $n = 2$ experiments (see Section 2.5) have shown that the algorithm typically converges in a small number of iterations (e.g., 5 – 15 iterations). Moreover, it can be shown that the original (observed data only) likelihood function is monotonically increased at each iteration of the algorithm, and the algorithm converges to a local maximum of the likelihood function.

2.4 Queuing Analysis of Back-to-Back Losses

Suppose that a pair of closely time-spaced packets arrives a finite length queue. Let $K$ denote the length of the queue and $k_0$ the number of packets in the queue immediately before (no intervening services or arrivals) the first packet in the pair arrives. If $k_0 = K$, then the packet is dropped (e.g., droptail assumption). The probability of $k_0 = j$, $j = 0, \ldots, K$ is denoted by $p(j)$, and we define the success probability is $\alpha = \sum_{j=0}^{K-1} p(j) = 1 - p(K)$. Between the arrival of the first and second packet several arrival and/or service events may occur. Let $r$ denote the total number of these events (both arrivals and services), and let $\ell_r$ denote the length of the queue immediately before the second packet arrives. Note that $\ell_0 = \min(k_0 + 1, K)$. We consider two possible conditional success probabilities.

Causal conditioning:

Probability that second packet successfully enters the queue, conditioned on the event that the first packet entered the queue. $\beta_r \equiv p(\ell_r < K \mid k_0 < K)$
Anti-causal conditioning:

Probability that the first packet successfully enters the queue, conditioned on the event that the second packet entered the queue. \( \gamma_r \equiv p(k_0 < K \mid \ell_r < K) \)

Previous work (7) has focused on the causal conditional probability, and follow-up work (34) has shown that \( \beta_r \approx 1 \) for inhomogeneous traffic (inhomogeneous Poisson arrival process), which may be typical in many cases. However, if the traffic is homogeneous traffic (fixed-rate Poisson arrival process), then \( \beta_r < \alpha \leq 1 \), which can degrade the performance of loss estimation algorithms. Although homogeneous traffic is rarely observed in practice, this undesirable property of causal conditioning motivates our investigation of the anti-causal case.

Notice, however, that for inference purposes, there is no problem with anti-causal conditioning, since we are interested in determining the success/loss rates responsible for a set of previous observations (i.e., our inferences are made after the actual measurements are taken). In this work, we emphasize the anti-causal conditional probability \( \gamma_r \) instead, for the following reasons. It is guaranteed that for all possible traffic models: (a) \( \gamma_0 = 1 \), and (b) \( \gamma_r \geq \alpha \), for all \( r \). Neither condition is met by \( \beta_r \) in general. In fact, for all cases considered in (34), \( \beta_r < 1 \) for all \( r \). Thus, it appears that \( \gamma_r \) is a much more suitable choice for network loss tomography, as it generally provides a higher level of conditional success probability and requires no assumptions on the traffic behavior. We summarize the results in the following theorem.
Theorem 2.1.

\[ p(k_0 < K \mid \ell_r < K) = \frac{1}{1 + c_r}, \]

where

\[ c_r = \frac{p(k_0 = K, \ell_r < K)}{p(k_0 < K, \ell_r < K)}. \]

Furthermore, \( c_0 = 0 \) and \( c_r \leq \frac{p(K)}{1 - p(K)} \) for all \( r \), which implies conditions (a) and (b) above.

Proof of Theorem 2.1 First note that

\[
p(k_0 < K \mid \ell_r < K) = \frac{p(k_0 < K, \ell_r < K)}{p(\ell_r < K)} \]
\[
= \frac{p(k_0 < K, \ell_r < K)}{p(\ell_r < K \mid k_0 < K)p(k_0 < K) + p(\ell_r < K \mid k_0 = K)p(k_0 = K)} \]
\[
= \frac{1}{1 + c_r}.
\]

Now observe that if \( n = 0 \) (no intervening events between packets), then \( p(k_0 = K, \ell_r < K) = 0 \) and thus \( \gamma_0 = 0 \). Also note that as \( r \) tends to infinity the effect of conditioning diminishes, and we have \( c_r \to \frac{p(K)}{1 - p(K)} \). Thus, \( \gamma_0 = 1 \) and \( \rho_\infty = 1 - p(K) = \alpha \), the unconditional success probability. Next, re-express \( c_r \) as

\[
c_r = \frac{p(\ell_r < K \mid k_0 = K) p(k_0 = K)}{p(\ell_r < K \mid k_0 < K) p(k_0 < K)} = \frac{1 - p(\ell_r = K \mid k_0 = K)}{1 - p(\ell_r = K \mid k_0 < K)} \frac{p(K)}{1 - p(K)}.
\]

To show that \( c_r \leq \frac{p(K)}{1 - p(K)} \), we simply need to verify that \( \frac{1 - p(\ell_r = K \mid k_0 = K)}{1 - p(\ell_r = K \mid k_0 < K)} \leq 1 \). To see that this is so, consider the probabilistic behavior of the queue after the event \( k_0 = K \).
or \( k_0 < K \). There are \( r \) (random) arrival and/or service events prior to the arrival of the second packet in the pair. Consider a realization of \( r \) such events. If, under \( k_0 < K \) we find that \( \ell_r = K \), then we can be certain that \( \ell_r = K \) under \( k_0 = K \) as well. This is true since if the number of arrivals outweigh the number of services in the former case, then starting with a full queue \( (k_0 = K) \) will lead to the same result. On the other hand, \( \ell_r = K \) given that \( k_0 = K \) does not imply that \( \ell_r = K \) given \( k_0 < K \). Because this holds for every realization, \( p(\ell_r = K \mid k_0 = K) \geq p(\ell_r = K \mid k_0 < K) \) or conversely \( p(\ell_r < K \mid k_0 = K) \leq p(\ell_r < K \mid k_0 < K) \), which finishes the proof.

The theorem establishes that \( \gamma_r \geq \alpha \) for all \( r \), regardless of the specific traffic and queueing behavior. If one makes further assumptions on the queueing behavior (e.g., an M/M/1/K model), then it is possible to determine the rate at which \( \gamma_r \) decays from 1 to the steady-state value \( \alpha \).

### 2.5 \texttt{nS-2} Simulation Experiments

We evaluated the passive loss inference framework using the \texttt{nS-2} simulation environment. In the simulations that we perform, we strive to investigate a number of issues. We gauge the performance of the combined EM loss inference algorithm and passive monitoring scheme under a variety of traffic conditions and queueing policies. We also explore the measurement period required to collect a sufficient number of data for accurate inference in a passive framework.
2.5.1 Simulation Framework

Network Topology: We use the same 12-node network topology in all experiments (see Figure 1.1). This topology is intended to reflect (to some extent) the nature of many networks — a slower entry link from the source, a fast internal backbone, and then slower exit links to the receivers. The chosen topology gives us the flexibility to explore the effects of having receivers at different distances from the source, and to examine the effect of varying fan-outs. We fix the size of all queues to be 35 packets. We consider four different traffic scenarios and perform all experiments twice, once using the droptail queuing policy throughout the network, and once using the Random-Early-Detection (RED) policy throughout.

Traffic Generation and Statistics Collection: In all experiments, we assume that there are TCP connections to the receivers that last for the extent of the measurement period. In addition, we set up a variety of short-duration TCP sessions, both from source to receiver and as cross-traffic on internal links, as well as exponential on/off traffic sources traversing various paths. In total there are approximately thirty TCP connections and thirty UDP connections operating within the network at any one time. The average utilization of the network is in all cases relatively high; otherwise, we experience many drops and loss estimation is of little interest.

We utilize all the TCP connections flowing from the source to the receivers when collecting statistics using the procedure discussed in Section 2.2. We set the maximum time-
spacing between packets within a pair to $\delta_t = 1$ ms and the minimum spacing between pairs to $\Delta_t = 10$ ms. We collect measurements over a 300 second interval. The four traffic scenarios are described as follows.

**Traffic Scenarios 1–3 (heavy losses at 1 or 2 links):** In these three traffic scenarios, we strive to ascertain the capability of our passive framework to discern where significant losses are occurring within the network. We assess its ability to determine how extensive the heavy losses are and to provide accurate estimates of loss rates on the better performing links. In each case, we establish each heavy loss link by adding substantial exponential on-off and short-duration TCP session cross-traffic flows to the link traffic. In Scenario 1, link 4-8 experiences heavy losses (enabling assessment of the framework’s ability to localize losses at links near the receivers). In Scenario 2, links 1-2 and 2-5 experience substantial losses (testing the framework’s capacity to separate cascaded losses). In Scenario 3, links 1-2 and 4-8 experience substantial loss. This last scenario tests the ability to resolve distributed losses in different branches of the network.

**Traffic Scenario 4 (mixed traffic with medium losses):** In this last scenario, we introduce many on-off UDP and on-off TCP connections throughout the topology and insert extra links (not depicted in Figure 1.1) connecting to the internal nodes. These links allow us to develop TCP cross-flows that have a range of different round-trip-times.
Figure 2.1: Simulation Results. True and estimated link-level success rates of TCP flows from source to receivers for all traffic scenarios. In each subfigure, the three panels display for each link 1 – 11 (horizontal axis): (top) true and estimated success rates using drop-tail queues, (middle) true and estimated success rates using RED queuing policy, and (bottom) mean absolute error for each link (RED and drop-tail).
Figure 2.2: The performance error (mean absolute error averaged over all links) versus measurement period.

2.5.2 Simulation Results

We conduct ten independent simulations of each traffic scenario and queuing policy over a measurement period of 300 seconds. Figure 2.1 displays the results of our simulations for each of the different traffic scenarios. We see that the estimated success rates are in good agreement with the true TCP success rates (based on direct counts of total losses on each link). In the heavy-traffic scenarios, we see that the worst-case mean absolute error is about one percent. The passive framework is thus capable of identifying where heavy losses occur. We do observe (see Figure 2.1 (d)) very slightly worse performance with RED in some cases, which is expected, since RED occasionally reduces the degree of loss correlation.

In Figure 2.2, we examine the relationship between the average mean absolute error and the measurement period. As expected, the error decreases as the measurement period...
increases. Note that even for a 60 second measurement period, the averaged mean absolute error is less than 0.6%.

Finally, we draw attention to the fact that the losses we measure for the TCP flows can be very different from those of the UDP traffic. For example, in the mixed traffic scenario, we observed average TCP losses of approximately 3% on links 6 and 10, whereas the on-off exponential traffic experienced losses of nearly 20%. This is a strong indication that active probing may provide a poor indication of losses in the existing TCP traffic.

2.6 Discussion

The sampling (or perhaps more aptly “mining”) of the TCP traffic flows for packet-pair events is a crucial step in our methodology. As discussed in Section 2.2, the simplest approach is simply to scan the traffic flows, locating packet-pair occurrences in a sequential fashion (beginning at the start of the measurement period). When a packet pair is located, we skip ahead $\Delta_\tau$ seconds and resume the scan. This results in a fast extraction algorithm, with the amount of measurements involving each receiver dependent on the throughput to that receiver. The disadvantage of this simple algorithm is that some pairs are more informative than others. Due to the nature of TCP, there are large numbers of back-to-back pairs to the same receiver (arising whenever the source receives notification that it can send a group of packets equal to the current window size). The pairs involving different receivers are rarer, occurring when the source switches from one connection to another. Because they are fewer in number, they provide more information for inference.
A potentially more effective alternative algorithm is to consider the reduced set of back-to-back "cross-pairs" that involve different receivers. We begin by locating and including all cross-pairs. The time intervals between cross-pairs are often sufficiently large so that we can include all of them without violating the requirement of a $\Delta_t$ separation. If not, we scan through the set; when we have to decide which of a set of closely-spaced cross-pairs to eliminate, we eliminate all but the pair with the least representation in the current set of included pairs. After the set of cross-pairs has been finalized, we incorporate "auto-pairs" (back-to-back packets destined for the same receiver), excepting those that violate the $\Delta_t$ time-separation criterion with the pairs already included. If we discover that this algorithm has resulted in the under-representation of some type of auto-pair, then we start again, including the under-represented pair in the initial set. Finally, single (isolated) packets are included, again maintaining the $\Delta_t$ time-separation. In this way, we can extract more informative statistics from a given set of traffic data, which should lead to improved
estimates. Conversely, a more effective sampling strategy of this nature should reduce the measurement time duration required for accurate loss inferences.

A more aggressive approach to obtain more informative data could involve alternative servicing strategies at the source. For example, instead of a basic round-robin service strategy, the source could employ a scheme that would enhance the chances of cross-pairs occurrences, without necessarily deviating from a TCP format.

One final point of discussion is the issue of scalability. The EM algorithm itself is reasonably scalable (approximately linear in the number of nodes), but as the number of receivers grows, the potential for a sufficient number of cross-pairs may diminish. Alternative data collection and servicing strategies, like those mentioned above, could mitigate this problem. Nonetheless, the possibility of "data-starvation" may limit one's ability to passively estimate all link-level losses in very large networks. This may not be as bad as it seems. For example, rather than estimating all link-level loss rates, in many practical situations it may be sufficient to determine loss rates on a few of the first links along the paths from the source to the receivers. By grouping receivers into clusters, the actual network can be abstracted into a smaller network with "effective nodes" replacing the original clusters. This is illustrated in Figure 2.3, where we use a "cloud" to denote the corresponding aggregation of links (subnetwork) to the clustered receivers. Auto-pairs and cross-pairs can be shared among clustered receivers, and we should be able to reliably estimate the loss rates on the upper links (close to the source) as well as average loss rates for the subnetworks associated with the clustered receivers.
2.7 Related Work

The techniques and performance analysis presented in this chapter are important extensions of the work presented in (7; 10; 18), but the basic idea of exploiting correlations of closely-spaced packets remains the same. The authors of (35) presented extensive performance analysis of the multicast loss inference technique proposed in (17); in this work, we aim to perform a similar analysis for passive unicast inference. In (7; 18) the nature of the probing (active vs. passive) was not specifically addressed, and the methodology proposed in (10) is based on active probing. The authors of (2) also investigated the problem of inferring packet losses from packet traces measured from Microsoft web servers.

2.8 Summary

The new passive unicast network tomography methodology we have proposed shows considerable promise. We have demonstrated using extensive ns-2 simulations that sufficient data can be collected using passive sampling to perform accurate loss inference, even for relatively short measurement periods. Moreover, we have observed that we are able to accurately estimate the losses experienced by existing TCP flows. As these can differ substantially from losses suffered by other forms of traffic, we surmise that in some situations inference from active probing may offer a poor reflection of existing TCP loss rates. We have conducted theoretical queuing analysis which corroborates the results obtained through simulation experiment.
Chapter 3

Nonparametric Network Delay Tomography

In this chapter, we examine the problem of inferring link-level delay statistics, using a non-parametric approach, from active measurements. We concentrate on networks comprised of a single source transmitting measurement probes to multiple receivers. The measurement framework is similar to the loss inference problem studied in the previous chapter. There is also no difficulty extending the approach to measurements made at multiple sources, although care must be taken that measurements are sufficiently separated for independence assumptions to hold. We assume that the routing (thus topology) is fixed throughout the measurement period, but straightforward extensions can account for changes in topology over coarse time scales. The assumption of fixed topology implies every probe packet sent to a specific receiver traverses the same path; i.e., the routes are unique, there are no route changes during the measurement period nor load-balancing in the routers.

The remainder of the chapter is structured in the following manner. In Section 3.1 we describe the measurement framework, modeling assumptions and implementation requirements. In Section 3.4 we describe the delay distribution inference methodology, detailing the Multiscale Maximum Penalized Likelihood Estimator (MMPLE) procedure and Expectation-Maximization (EM) algorithm. In Section 3.5 we describe the results of ns-2 experiments designed to explore the performance of the methodology. In Section 3.6, we
describe some of the related works. In Section 3.7, we make some concluding remarks.

3.1 Measurement Framework

For the networks we consider, standard network routing protocols force packets to follow a specific route indicated by the routing table and they produce a tree-structured logical topology (see Chapter 1, with the source at the root and the receivers at the leaves). An example of network with six receivers under study in this chapter is depicted in Fig. 3.1. We adopt the notation that link \( i \) connects node \( i \) (below) to its parent node (above) and traffic is flowing from the parent to the node. We consider the situation where measurements can only be made at the edge of the network and assume that the routing table (and thus topology) is fixed and known for the duration of the measurement.

![Tree-structured network topology](image)

**Figure 3.1** : Tree-structured network topology used for ns-2 simulation experiments. Source (node 0) transmits to 6 receivers (nodes 6 – 11). Link speeds in Mb/s are shown next to the links. Link \( i \) connects node \( i \) to its parent node, e.g. link 9 connects nodes 5 and 9.

The basic measurement and inference idea is quite straightforward. Suppose two closely
time-spaced (back-to-back) packets are sent from the source to two different receivers. The paths to these receivers traverse a common set of links, but at some point the two paths diverge (as the tree branches). The two packets should experience approximately the same delay on each shared link in their path. This facilitates the estimation of the delays occurring on each link.

We distinguish between a measurement period and an inference period. The measurement period is the time period over which all measurements are collected. The inference period is some time window within the measurement period; the window duration is dictated by the degree of network stationarity and only the measurements collected in this window are used to perform inference. In order to achieve estimates over the entire measurement period, multiple inferences must be performed using different, potentially overlapping inference windows.

We collect measurements of the end-to-end delays (sum of propagation, transmission and queuing delays) from source to receivers, and we index the packet pair measurements by $k = 1, \ldots, N$. For the $k$-th packet pair measurement, let $y_1(k)$ and $y_2(k)$ denote the two end-to-end delays measured. The ordering 1 and 2 is arbitrary; the indices are randomly selected with no dependence on the order in which the packets were sent from the source. This will be important in dealing with discrepancies between the delays experienced by the two packets on shared links, which will be discussed in greater detail in Section 3.2. In this work, we do not consider the case in which one or both of the packets is dropped (lost). We simply discard packet pairs in which a loss occurs. However, it is possible to extend
our approach to include losses. Since we are interested in inferring queuing delay, our first step is to extract what we perceive as the minimum delay (propagation + transmission) on each measurement path. The minimum delay corresponds to the case in which all queues in the path are empty (i.e., no queuing delay). This is estimated as the smallest delay measurement we acquire on the path during the measurement period. We assume that the true minimum delay is observed over the measurement period. If this is not the case, then queuing delay is systematically underestimated for links on the affected path.

Our goal is a nonparametric estimate of the delay distributions on each link. Clearly it is impossible to completely determine an infinite dimensional density function from a finite number of delay measurements, but we require that as the number of delay measurements increases so does the accuracy of our estimation procedure. Thus, we adopt the following procedure. The end-to-end delay measurements are binned, but the number of bins is chosen to be equal to or greater than the number of delay measurements. We stress that this is not a parametric step. This means that there is less than one measurement per bin, on average, and hence we do not lump or group delays in an artificial, prescribed fashion. Thus, we place no prior restriction on the form of the density estimator; the more measurements one has, the more one can resolve the structural nuances of the delay densities.

In practice we choose the number of bins to be the smallest power of two greater than or equal to the number of measurements (facilitating certain processing steps to be described later). We upper bound the maximum delay on any one link by the maximum end-to-end delay along the path(s) that include the link. Let $d_{\text{max}}$ denotes the maximum path delay
on any link. This upper bound for a particular link. Let K be the smallest power of 2 that is greater than or equal to the number of measurement packets N. The bin width for the link is then set at \( d_{\text{max}}/(K - 1) \). This procedure is conservative, in that the estimated \( d_{\text{max}} \) may be substantially larger than the true maximum queuing delay. It may be preferable to use previous link-delay estimates or bandwidth estimates from a procedure such as nettimer (14) to gauge the maximum delay on any link.

At this stage, each end-to-end measurement has been ascribed a discrete number between 0 and \( (K - 1) \). To illustrate our inference methodology in its simplest form, suppose that we send many packet pairs to receivers 6 and 7 in Fig. 3.1 and measure the delays experienced by each packet. Each measurement consists of a pair of delays, one being the delay to receiver 6 and the other the delay to receiver 7. From these measurements, collect events where ‘0’ delay (a delay in bin zero) is measured at receiver 6. Now, assuming that the delay is the same for both packets on the common links (1 and 2 in this case), any “additional” delay observed to the receiver at 7 can be attributed to link 7 alone. We can then build a histogram estimate of the delay pmf for link 7. This simple idea can be extended and improved to obtain estimators for the delay distributions on all links which take advantage of all the measured data (not just special cases like the one above). In Section 3.4, we describe the large-scale inference procedure in detail.

The basic inference idea is simple. Suppose the network is stationary over each measurement period, the delays are identical on shared links, and the true delay pmfs are strictly positive and canonical (there is some mass in the zero delay bin). This implies the first
packet has left the queue before the second packet enters. The delay experienced by the second packet will not be dependent on the delay of the first one. In addition, suppose that the link delays experienced by an individual packet are independent of one another as in the multicast scenario. Then, based on the identifiability analysis carried out for the multicast case (16), one can easily show that the true distributions can be uniquely identified from such end-to-end measurements (as the number of measurements tends to infinity). The issue about slightly different delays on shared link in practice will be addressed in the following section. It is important to point out that unique identification is not possible (in general) using single packet delay measurements; there are ambiguous cases that cannot be resolved without multiple packet correlations (7).

3.2 Model Assumptions

There are several assumptions in the framework that are worthy of discussion. Firstly, we assume spatial independence of delay. Delay on neighboring links is generally correlated to a greater or lesser extent depending on the amount of shared traffic. In the ns-2 (19) experiments discussed in Section 3.5, weak correlation of delays is observed. In the presence of weak correlation, our framework is able to derive good estimates of the delay distributions. As the correlation grows stronger, we see a gradual increase of bias in the estimates. We also assume temporal independence (successive probes across the same link experience independent delays). Temporal dependence was observed in (16) and in our experiments; indeed it is exploited in (9). As in (16), the maximum likelihood estimator we employ re-
mains consistent in the presence of temporal dependence, but the convergence rate slows.
In practical situations, dependencies are usually weak and do not have a dramatic effect
on the performance of the estimator. Ignoring dependencies can also be interpreted and
analyzed as a case of Besag’s pseudo-likelihood approach (36).

Finally, our framework hinges on an assumption that packets in a pair experience a com-
mon delay on shared links. If the delays are identical on shared links, then the difference
between the two delay measurements can be attributed solely to the delays experienced on
unshared links in the two paths. This is the key to uniquely determining the delays on a
link-by-link basis. However, in practice the two packets may experience slightly different
delays on shared links due to the fact that one packet precedes the other in the common
queues and additional packets may intervene between the two. The nature of this delay
differential is exposed in Fig. 3.2, which shows the histogram of the difference between the
end-to-end delays of two closely-spaced packets sent to the same receiver over the Inter-
et. This histogram is constructed from back-to-back packet pair measurements using the
netdyn tool (21). Ideally, the delays should be identical, but we see a small discrepancy
between the two. The second packet in the pair typically experiences a slightly greater de-
lay. However, recall that the ordering of the packets was arbitrary in our recording process.
In effect then, the discrepancies between the delays on shared links adds an approximately
zero mean error to the difference between the two end-to-end measurements. We clearly
see the symmetric zero-mean nature in the empirical data shown in Fig. 3.2, and we have
observed similar behavior in all our measurements and simulations. This “noise” produces
a smoothing (or blurring) in the inferred delay pmfs. Nonetheless, because the errors are roughly zero mean, we can still use the estimated delay pmfs to obtain approximately unbiased estimates of the expected delay (Fig. 3.2b) on each link or the locations of modes in the density, for example. The errors could also be directly modeled, but our experimentation suggests that these errors are relatively insignificant in the overall process, due to the greater variability caused by the limited number of probes that can be used in practical situations.

![Figure 3.2](image)

(a) End-to-end delay histogram (packets sent from Rice University to Michigan State University). (b) Difference between delays of the two packets in packet pairs. Measurements were made using the netdyn tool.

### 3.3 Measurement Requirements

The delay inference framework requires knowledge of the (logical) topology of the network and the capability to perform one-way delay measurements. We perform the construction
of the topology using a modified, lightweight version of traceroute (37; 38). Alternatively, it is possible to determine the topology using the end-to-end unicast measurement and inference procedure we recently proposed in (4). Collection of one-way delay measurements requires that the receivers cooperate with the source and the precision of the system timing (39).

We do not necessarily require that clocks at the source and receivers be synchronized, but we do require that the disparity between clocks remain very nearly constant over the measurement period. In this way we can be sure that subtracting the estimated minimum delay does not induce bias in our estimates. A further difficulty lies in clock resolution. Clocks must be precise enough to ensure that time measurement errors are insignificant relative to the scale of the time delays of interest. Deployment of Global Positioning System (GPS) devices allows these clock difficulties to be avoided, as it provides synchronized measurements to within tenths of microseconds. Furthermore, High Definition Television (HDTV) signals could potentially be useful for high precision indoor time measurements (up to microsecond resolution). Alternatively, delay measurements can be adjusted using algorithms developed to detect and compensate for clock adjustments and rate discrepancies (39; 40; 41). In this work, we assume that synchronized measurements are available.

3.4 Delay Distribution Inference

We commence with the description of our inference framework by formalizing our measurement and modeling notation. Let $p_i = \{p_{i,0}, \ldots, p_{i,K-1}\}$ denote the probabilities of
a delay of $0, 1, \ldots, K - 1$ time units, respectively, on link $i$. We denote the packet pair measurements $\mathbf{y} \equiv \{y_1(k), y_2(k)\}_{k=1}^N$.

In general, only a relatively small amount of data can be collected over the period when delay distributions can be assumed approximately stationary. A natural estimate would be the maximum likelihood estimates (MLEs) of $\mathbf{p} \equiv \{p_i\}$, the collection of all delay pmfs. However, if a large number of bins is used (i.e., high-resolution delay estimates), then the problem is ill-posed and the MLE tends to overfit to the probe data (see Fig. 1.2(a)), producing highly variable estimates that do not accurately reflect the delay distribution of the traffic at large. High variance manifests itself in irregular, noisy-looking estimates (42). One way to reduce this irregularity is to maximize a penalized likelihood (see Fig. 1.2(d)). We replace the maximum (log) likelihood objective function $L(\mathbf{p}) = \log l(\mathbf{y}|\mathbf{p})$ with an objective function of the form:

$$L(\mathbf{p}) - \text{pen}(\mathbf{p})$$

where $\text{pen}(\mathbf{p})$ is a non-negative real-valued functional that penalizes the irregularity (or complexity) of $\mathbf{p}$. A small value of $\text{pen}(\mathbf{p})$ indicates that $\mathbf{p}$ is a smooth, regular function; a large value indicates that $\mathbf{p}$ is irregular and complex function. The maximization of this penalized log-likelihood involves a trade-off between fidelity to the data (large $L(\mathbf{p})$) and smoothness or simplicity (small $\text{pen}(\mathbf{p})$). We will describe a specific choice of penalty functional in Section 3.4.2. Before moving to that, however, we will quickly formulate the basic likelihood function and motivate the adoption of an Expectation-Maximization (EM)
algorithm for optimization.

3.4.1 Likelihood Function

Under the assumption of spatial independence, the likelihood of each delay measurement \( \{y_1(k), y_2(k)\} \) is parameterized by a convolution of the pmfs in the path from the source to receiver. With our modeling constraint that packets in a pair experience the same delay on shared links, the likelihood of the two measurements made by the \( k \)-th packet pair is:

\[
l(y_1(k), y_2(k)|p) = \sum_j \rho_{c,k}(j) \rho_{1,k}(y_1(k) - j) \rho_{2,k}(y_2(k) - j). \tag{3.2}
\]

The range of the summation is determined by the ranges of the pmfs \( \rho_{c,k}, \rho_{1,k} \) and \( \rho_{2,k} \). The pmf \( \rho_{c,k} \) is the convolution of the pmfs of the links on the shared path of the two packets, e.g. \( \rho_{c,k} = p_1 * p_2 \) for a \( 6 - 7 \) pair in Fig. 3.1 (with \( * \) denoting convolution). The pmf \( \rho_{1,k} \) (resp. \( \rho_{2,k} \)) is the convolution of the pmfs on the links traversed only by the packet that measures \( y_1(k) \) (resp. \( y_2(k) \)). The joint likelihood \( l(y|p) \) of all measurements is equal to a product of the individual likelihoods:

\[
l(y|p) = \prod_{k=1}^{N} l(y_1(k), y_2(k)|p). \tag{3.3}
\]

The presence of convolved link pmfs in the likelihood of each measurement (Equation 3.2) results in an objective function that cannot be maximized analytically. The max-
imization of the likelihood function requires numerical optimization, and an Expectation-Maximization (EM) algorithm (43) is an attractive strategy for this purpose. Before giving the details of the algorithm, we briefly review the multiscale maximum penalized likelihood estimate (MPMLE) nonparametric density estimation procedure employed in our framework.

3.4.2 MIMPL Density Estimation

Here we briefly outline the MIMPLE density estimation procedure developed in (25; 26).

To introduce the idea, we consider a case where the link delays have been directly measured. Let $z_i(k)$, $k = 1, \ldots, N_i$, denote a set of delay measurements for a particular link $i$. We assume that these measurements are independent and identically distributed according to a continuous delay density $p(t)$, where without loss of generality we assume that $t \in [0, 1]$ (for convenience of exposition we take the maximum delay to be unity).

Define a discrete pmf via $p_{i,j} = \int_{(j)/K}^{(j+1)/K} p(t)dt$, $j = 0, \ldots, K - 1$, where $K$ is the smallest power of two greater than or equal to $N_i$. It follows that the number of measurements falling in the interval $[\frac{j}{K}, \frac{(j+1)}{K}]$, denoted $m_{i,j}$, is multinomially distributed (22), i.e., $\{m_{i,j}\} \sim \text{Multinomial}(N_i; \{p_{i,j}\})$. The MIMPLE estimator maximizes the following criterion with respect to $\{p_{i,j}\}$:

$$\log \text{Multinomial}(N_i; \{p_{i,j}\}) - \text{pen}(\{p_{i,j}\}),$$

(3.4)
where

\[ \text{pen}(\{p_{i,j}\}) \equiv \frac{1}{2} \log(N_i) \times \#_i, \]

(3.5)

where \#_i is the number of non-zero coefficients in the discrete Haar wavelet transform of the pmf \{p_{i,j}\}. This number reflects the irregularity and complexity of the pmf — the larger the value of \#_i, the more “bumps” in the pmf. There are two important features of the MMPL: (1) the global maximizer can be computed in \(O(K)\) operations; (2) the MMPL is nearly minimax optimal in the rate of convergence over a broad class of function spaces (25; 26).

Computing the MMPL is very similar to standard wavelet denoising methods. Finding the optimal solution to (3.4) involves computing the Haar wavelet transform of the pmf and thresholding (“keeping” or “killing”) each Haar wavelet coefficient according to a generalized likelihood ratio test (GLRT). Due to the multinomial form of the likelihood, the GLRTs involve binomial statistics (instead of the usual Gaussian statistics involved in standard wavelet denoising problems). The physical interpretation of each GLRT is simple: if the magnitude of the wavelet coefficient is sufficiently large, then that coefficient is left unaltered, otherwise it is set to zero. In detail, the MMPL estimator is computed according to the four steps below.

1. Compute the (unnormalized) Haar scaling coefficients of the sequence \{m_{i,j}\} as follows.
For scales $\ell = 0, \ldots, \log_2 N_i$

$$s_{i,j}^\ell = \sum_{j=1}^{2^\ell} m_{i,j+k2^\ell}, \quad k = 0, \ldots, N_i/2^\ell - 1.$$ 

Note that $\{m_{i,j}\}$ are the scaling coefficients at scale $\ell = 0$.

\(\text{ii. Form the “multiscale coefficients”}\)

$$\rho_{i,j}^\ell = \frac{s_{i,2j}^{\ell-1}}{(s_{i,2j}^{\ell-1} + s_{i,2j+1}^{\ell-1})}.$$ 

Note that $s_{i,j}^\ell = s_{i,2j}^{\ell-1} + s_{i,2j+1}^{\ell-1}$. Therefore, the scaling coefficients at scale $\ell - 1$ can be constructed from the scaling coefficients at scale $\ell$ along with the multiscale coefficients at scale $\ell$ according to

$$s_{i,2j}^{\ell-1} = \rho_{i,j}^\ell s_{i,j}^\ell \quad \text{and} \quad s_{i,2j+1}^{\ell-1} = (1 - \rho_{i,j}^\ell) s_{i,j}^\ell. \quad (3.6)$$

The multiscale coefficients are closely related to the usual Haar wavelet coefficients.

Specifically, the (unnormalized) Haar wavelet coefficient is

$$\omega_{i,j}^\ell = s_{i,2j}^{\ell-1} - s_{i,2j+1}^{\ell-1}$$

$$= (2\rho_{i,j}^\ell - 1)s_{i,j}^\ell.$$ 

Note, in particular, that if $\rho_{i,j}^\ell = 1/2$, then $\omega_{i,j}^\ell = 0$. 
iii. Compute the test statistic

\[ t_{i,j}^\ell = s_{i,2j}^{\ell-1} \left( \log(\rho_{i,j}^\ell) - \log(1/2) \right) + s_{i,2j+1}^{\ell-1} \left( \log(1 - \rho_{i,j}^\ell) - \log(1/2) \right) , \]

and “threshold” the multiscale coefficients according to

\[ \delta(\rho_{i,j}^\ell) = \begin{cases} 
1/2, & \text{if } t_{i,j}^\ell < 1/2 \log N_i \\
\rho_{i,j}^\ell, & \text{if } t_{i,j}^\ell \geq 1/2 \log N_i 
\end{cases} \]

iv. Construct the MMPL estimate by recursively applying (3.6) beginning with \( s_{i,1}^{\log_2 N_i} = 1 \) and using the thresholded multiscale coefficients \( \{\delta(\rho_{i,j}^\ell)\} \) in place of the original coefficients. The resulting scale \( \ell = 0 \) scaling coefficients are the desired elements of the MMPL estimator \( \{\hat{p}_{i,j}\} \).

The near minimax optimality implies that the rate at which the estimator converges to the true continuous density (as a function of the number of measurements \( N_i \)) cannot be significantly improved upon. More complicated and computationally intensive procedures will not significantly outperform the MMPL. The optimization is carried out by performing a set of \( K \) independent generalized likelihood ratio tests. In all results in this chapter we employ a translation-invariant version of the MMPL, in which multiple MMPLs are computed with \( K \) different shifted versions of the Haar wavelet basis and the resulting estimates are averaged. This produces a slight improvement over the basic MMPL and
can be efficiently computed in $O(K \log K)$ operations.

### 3.4.3 EM Algorithm

The MMPLE methodology can be employed in the tomographic delay estimation case by simply adopting the penalized likelihood criterion:

$$\log l(y|p) - \sum_i \frac{1}{2} \log(N_i) \times \#_i, \quad (3.7)$$

where $N_i$ denotes the number probe packets passing through link $i$ and $\#_i$ denotes the number of non-zero Haar wavelet coefficients in the delay pmf $p_i$ of link $i$. Unfortunately, the penalized likelihood function cannot be maximized analytically due to the convolutional relationship between link delay pmfs and end-to-end measurements $y$.

The first step in developing an EM algorithm is to propose a suitable *complete data* quantity that simplifies the likelihood function. Let $z_i(k)$ denote the delay on link $i$ for the packets in the $k$-th pair. Let $z_i = \{z_i(k)\}$ and $z = \{z_i\}$. The link delays $z$ are not observed, and hence $z$ is called the *unobserved data*. Define the *complete data* $x \equiv \{y, z\}$. Note that the complete data likelihood may be factorized as follows:

$$l(x|p) = f(y|z)g(z|p),$$

where $f$ is the conditional pmf of $y$ given $z$ (which is a point mass function since $z$ deter-
mines \( y \), and \( g \) is the likelihood of \( z \). The factorization shows that \( l(x|p) \propto g(z|p) \), since \( f(y|z) \) does not depend on the parameters \( p \). Next note that the likelihood

\[
g(z|p) = \prod_{i,j} p_{i,j}^{m_{i,j}},
\]

where \( m_{i,j} \equiv \sum_{k=1}^{N} 1_{z(k)=j} \) is the number of packets (out of all the packet pair measurements) that experienced a delay of \( j \) on link \( i \); here \( 1_A \) denotes the indicator function of the event \( A \). Therefore, we have

\[
l(x|p) \propto \prod_{i,j} p_{i,j}^{m_{i,j}},
\]

and, if the \( m_{i,j} \) were available, then the MLE of \( p_{i,j} \) would be simply

\[
\hat{p}_{i,j} = \frac{m_{i,j}}{\sum_{k=0}^{K-1} m_{i,k}}.
\] (3.8)

Similarly, given the \( m_{i,j} \) we could directly apply the MMLE described above (see (25; 26; 44) for implementation details).

The EM algorithm is an iterative method that constructs and utilizes a complete data likelihood function to maximize the original likelihood function. By suitable modification, it can be used to maximize a penalized log-likelihood objective function like (3.7), whilst preserving the advantage of the O(K) computational simplicity of the MMLE technique.

When a modified EM algorithm is used to maximize a penalized log-likelihood function, it alternates between computing the conditional expectation of the complete data log
likelihood given the observations $y$ and maximizing the sum of this expectation and the imposed complexity penalty ($-\text{pen}(p)$) with respect to $p$. Notice that, ignoring constant terms, the complete data log likelihood is linear in $m$:

$$\log l(x|p) \propto \sum_{i,j} m_{i,j} \log p_{i,j}.$$ 

Thus, in the E-Step we need only compute the expectation of $m = \{m_{i,j}\}$.

**E-Step:** Let $p^{(r)}$ denote the value of $p$ after the $r$-th iteration. Then

$$\hat{m}_{i,j}^{(r)} = E_{p^{(r)}}[m_{i,j}|y],$$

$$= E_{p^{(r)}} \left[ \sum_{k=1}^{N_i} 1_{\{z_i(k) = j\}} | y \right],$$

$$= \sum_{k=1}^{N_i} E_{p^{(r)}} \left[ 1_{\{z_i(k) = j\}} | y \right],$$

$$= \sum_{k=1}^{N_i} E_{p^{(r)}} \left[ 1_{\{z_i(k) = j\}} | y_1(k), y_2(k) \right],$$

$$= \sum_{k=1}^{N_i} p^{(r)}(z_i(k) = j | y_1(k), y_2(k)). \quad (3.9)$$

Thus, the conditional expectation of $m$ can be computed by determining the conditional probabilities above for each packet pair measurement. A fast message-passing algorithm for this calculation is described in the next section.
**M-Step:** In the penalized case (3.7), apply the MMLE algorithm described in Section 3.4.2 with the conditional expectation \( \{ \hat{m}_{i,j}^{(r)} \} \) in place of \( \{ m_{i,j} \} \). In the case of unpenalized maximum likelihood estimation, simply substitute \( \{ \hat{m}_{i,j}^{(r)} \} \) in place of \( \{ m_{i,j} \} \) in equation (3.8).

### 3.4.4 Fast Fourier Transform based EM Algorithm

The expectation step of the EM algorithm poses the major portion of the computational burden of the optimization task. It can be performed using a message passing (or upward-downward) procedure (45). The message passing procedure is based on a factorization of the likelihood function. According to (Equation 3.9), our task for each measurement in the \( r \)-th iteration of the EM algorithm is to compute \( p^{(r)}(z_i = j|y_1, y_2) \) (we have dropped the measurement index \( k \) for notational ease). In the 1980’s, Pearl (46) and Spiegelhalter (47) independently developed the message passing methodology, an exact probability propagation algorithm for inferring the distributions of individual variables in singly-connected graphical models (factor graphs). The basic idea of the algorithm is that each node in the graph propagates its information (a measurement or current pmf estimate in this case) to every other node. Each node then combines all the messages it receives to compute the distribution of its variable.

Figure 3.3 depicts an example of the type of graphical model that arises in the delay inference procedure. This factor graph is used for evaluation of the pmf estimates in the \( r + 1 \)-th iteration of the EM algorithm. In this factor graph, the nodes labeled \( d_i \) correspond
to the nodes of the tree that are involved in measurement to nodes 6 and 7 in the example network. The nodes $p_i^{(r)}$ contain the delay pmf estimates that were generated in the previous iteration of the algorithm. The nodes labeled $z$ represent the complete data, that is, the unobserved individual link delays.

![Factor Graph](image)

Figure 3.3: The factor graph used in the message-passing algorithm for a measurement made by a packet pair sent to nodes 6 and 7 in the network of Fig. 3.1. Measurements are available at nodes 6 and 7; the nodes $p_i^{(r)}$ contain current pmf estimates; and node $c_{a,b}$ indicates the convolutional relationship between nodes $d_a$, $d_b$, and $z_b$.

We will briefly illustrate the operation of this message passing algorithm by considering how it behaves when acting on a measurement made by a packet pair destined for nodes 6 and 7 in the example network. The message passing algorithm can be divided into two stages. In the upward stage, starting at the leaves, information is passed via messages from node to node until the root is reached. In the downward stage, information from the root is passed via messages from node to node until the leaves are reached. Individual nodes then
combine the upward and downward messages they received to generate marginal pmfs for their values.

At a leaf node ($d_6$ or $d_7$) in Figure 3.3, the upward message is simply a delay pmf that has a one in the bin of the delay measurement being processed and zeros everywhere else. The upward message from $z_6$ is the previous pmf estimate for link 6. At node $c_{2,6}$ this message is convolved with the message from the leaf node $d_6$, and the result is passed up to the branching point $d_2$. A similar process occurs from leaf node 7. At node $d_2$, the upward messages from the two lower branches are multiplied together and the resultant message is passed up. The convolution procedure continues up the shared branch until the root node is reached. In the downward stage, the initial message from the root contains the information that the delay at the root is zero: it is a delay pmf with one in the zero bin and zeros elsewhere. Messages are passed down, with convolution exactly as before. At the branching node $d_2$, the message passed down to node $c_{2,6}$ is the product of the downward message from $c_{1,2}$ and the upward message from $c_{2,7}$. At the end of the two stages, the each node $z_i$ multiplies the upward message, the downward message, and its distribution from the previous EM iteration to obtain $p^{(r+1)}_i(z_i = j | y_1(k), y_2(k))$.

A straightforward implementation of this message passing procedure, as first proposed in (8), has a computational complexity of $O(LK^2)$ per measurement and iteration of EM, where $L$ is the maximum path length in the network and $K$ is the number of bins. Recall that $K$ is the smallest power of two greater than or equal to $N$. For each measurement, the act of passing a message within the algorithm involves the evaluation of a number of
summations, which can be cast as convolutions. These convolutions involve vectors of maximum length $LK$, where $L$ is the maximum path length in the network. Implementation of the convolutions in the Fourier domain reduces the computational complexity from $O(LK^2)$ to $O(LK \log K)$ per measurement and iteration of EM. This reduction can be substantial when $N$ (and hence $K$) is reasonably large.

3.5 Simulation Experiments

In order to verify the performance of our estimation methodology, we conducted $n = 2$ (19) simulation experiments using the network depicted in Fig. 3.1. Interior links in the network have higher capacity (5-10 Mb/sec) and propagation delay (50 ms) than the edge links (0.5 – 2 Mb/sec and 10 ms). Queues are FIFO (droptail) with space for 35 packets. Node 0 generates a 19.2 Kbit/s probing stream comprised of UDP packet-pair probes (60 bytes each). Packet-pair sending times are generated according to a Poisson process; the mean time-spacing is 50 ms. The probe-stream requires less than 1% of any link’s capacity. Background traffic is composed of a mixture of long-lived data-source TCP (FTP) connections, exponential on-off sources using UDP, and multiple short-duration TCP connections. Averaged over the simulations, link utilization ranges between 10 and 60 percent, and loss rates ranged from 0 to 2%; typical values for certain real networks.

The network was simulated for multiple two minute measurement periods; from within each measurement period, 25 s (inference period) was isolated for analysis. This time duration corresponds to 500 packet-pairs (assuming no probes are lost). Throughout the in-
ference period, queue lengths in the network were determined at a fine time scale by monitoring the arrivals of every packet at each queue. A "true" pmf for each link was formed by calculating delays from queue lengths and link capacities, quantizing and forming a histogram. When generating this true pmf, so much data is available that the quantization can be very fine (constructing an excellent estimate of the delay density) without affecting estimation stability.

In Fig. 3.4, we show the results of one experiment, comparing the true pmfs to the nonparametric MMPLLE estimator and the MLE estimator of (8) using a 16-bin discretized pmf (16 bins was found to give the best performance among unpenalized estimators; see discussion below). We display results for the lower bandwidth links because for our experimental set-up, queuing delay was concentrated in these links. We display the results of representative links that provide a meaningful indication of performance. There is substantial mass in the tails of these pmfs and we can evaluate how well the pmf estimates generated by our proposed methodology estimates match the tails; network performance hinges critically on the tail probabilities of queues (48; 49). In the higher bandwidth links, there is much less mass in the pmf tails 3.7(a). For these links, both the MLE and MMPLLE estimates match the true pmf where probability mass is concentrated, but there is insufficient information to closely match the tails. We calculated the MLE for a variety of bin sizes, but show the bin size that achieved the best fit to the true pmf (in this case 16 bins). The nonparametric estimator was calculated from $K = 512$ bins.

In Fig. 3.5, we plot the magnitude of the $L_1$ error norm between the true pmf and the
Figure 3.4: Comparison between true pmfs (solid) and estimated pmfs (dashed). Top panel shows true pmf and MMPLSE (calculated using 512 bins); bottom panel shows true pmf and MLE (calculated using 16 bins). 16 bins is determined as the bin size at which the MLE obtains the best fit. (a) Link 5. (b) Link 7. (c) Link 9.

MMPLSE for the links in the network, as averaged over 25 simulations. Also shown are the results for the MLE for medium (64 bins) and large (16 bins) bin sizes. The $L_1$ error norm is simply the sum of the absolute difference between the estimated pmf and the true pmf over the K bins. As discussed in (22; 50), the $L_1$ error criterion is a common measure of the performance of a density estimate. The advantage of such as a measure as opposed to a mean-squared error criterion is that more attention is paid to the tails of the distributions. It also enjoys several theoretical advantages over other measures (50).

As is evident from the two figures, the MMPLSE technique generates estimates which are smooth, close fits to the true pmfs. In order to introduce some degree of smoothness, MLE estimates must be calculated using a large bin size, resulting in an inability to capture the finer details of a pmf.

In order to illustrate the performance of the algorithm in a larger network, we also simulate a 20-receivers scenario as shown in Fig. 3.6. The packet probing rate from the source,
as well as the composition of background traffic remains the same as in the first scenario. The link loss rates range from 0 to 2 percent, and the link utilization varies between 0 and 60 percent, averaged over 20 simulations. We use the same inference window of 25 seconds.

If we assume there is no packet loss, then there are a total of 500 packet pairs. However as the number of total measurements remains unchanged while the number of receivers increases, the number of measurements obtained for each link reduces. In Fig. 3.7 and in Fig. 3.8, we show the results and performance of the algorithm. Fig. 3.9 compares the delay cumulative distribution function (cdf) obtained by estimation based on direct measurement with the delay cdf estimated using the MMPLE technique and the probe measurements for a representative link in the network.

When the amount of probing that can be performed is limited, we believe that the most substantial source of error is the intrinsic variability in probe measurements. Another potential source of error is the discrepancy between the delays experienced by the two pack-
Figure 3.6: A larger tree-structured network topology used for ns-2 simulation experiments. Source (node 0) transmits to 20 receivers (nodes 19-38). Link speeds in Mb/s are shown next to the links.

ets in each pair on their common path. We therefore examined the extent and effect of the delay discrepancy; with 512 bins, the overwhelming majority of the discrepancy was concentrated in 0-3 bins, with a maximum value of 16 bins. The effect of these discrepancies on the quality of the estimates is relatively minor when such a small amount of data is available for inference. If we directly measure the delays experienced by probes on each link (which can be done in our simulation), the estimates we obtain are very similar to those obtained by our tomographic procedure.

3.6 Related Work

Lo Presti et al. have outlined a framework for the inference of internal queuing delay distributions based on multicast end-to-end measurement (16). Multicast-based procedures for estimating low order moments such as link delay variances have also been developed (51).
Figure 3.7: Comparison between true pmfs (solid) and estimated pmfs (dashed). Top panel shows true pmf and MMPL (calculated using 512 bins); middle panel shows true pmf and MLE (calculated using 64 bins); bottom panel shows true pmf and MLE (calculated using 512 bins). 512 bins is determined as the bin size at which the MLE obtains the best fit. (a) Link 1. (b) Link 20. (c) Link 31.

The multicast framework has the advantage of scalability (each measurement probe provides some information about all links in the considered network) and guaranteed, structured correlation between the delay measurements at different receivers. However, multicast is not supported by all networks and there is evidence that routers treat multicast packets differently from the unicast packets that make up the majority of network traffic (10). These concerns motivate the development of an inference framework based on unicast measurement. However, an important new consideration arises in the unicast setting. For a fixed measurement overhead, multicast measurement provides much more data than unicast. This means that if the framework of (16) were adapted to unicast measurement, as suggested in (10), it would need to perform with significantly less information available.

Lai and Baker (14) have implemented nettimer, a procedure that estimates link-level bandwidth. Similar in nature to pathchar (52), it exploits the time-to-live field
Figure 3.8: \( L_1 \) error criterion averaged over 20 simulations (means and standard deviation) for some terminating links. Solid line is MMPLE, dash-dot line is MLE (512 bins), and dotted line is MLE (256 bins).

of packets to collect informative measurements. nettimer generates accurate estimates of bandwidths (particularly when they are small), although it requires a relatively large number of measurement packets. Theoretically, it could be used to estimate queuing delays, but to our knowledge there has been no experimental work exploring its performance. The number of measurement packets needed for estimation may prove prohibitive given the short duration over which delay distributions are generally stable. It would seem that network utilization would need to be low in order to achieve reliable estimates.

Shih and Hero have developed a method for estimation of the link delay cumulant generating functions (CGFs) (53; 54). The CGFs have the advantage of being additive over a path of several links in contrast to the convolutional way in which link delay pmfs combine to form end-to-end delay pmfs. Based on the disentangled CGFs, it is straightforward to reconstruct the delay distributions. This technique has the benefit of imposing no dis-
Figure 3.9: The cdf estimates obtained from direct measurement (solid) to the tomographic one (dotted). (a) Link 1. (b) Link 20. (c) Link 31.

cretization, but does not impose smoothness constraints, leading to an ill-posed problem when data is limited. The chief disadvantage of the technique is that in order for all links to be resolved, internal measurements must be available or a tool such as nettimer must be used.

Coates and Nowak have described a sequential Monte Carlo-based internal delay estimation framework in (8; 9). This framework directly addresses the time-varying nature of network delay behavior. In this approach, a fine-level of quantization can be imposed, and smoothness is incorporated through the adoption of a slowly-varying time-dependent Bayesian prior. However, the parameters associated with the prior introduce a potentially undesirable parametric nature to the estimation task.

Anagnostakis and Greenwald have explored the feasibilities of using existing network infrastructure in making delay measurements (55; 56). They have also studied the differences in direct measurements and indirect inference for determining the internal delays. The direct measurements depend on the Time-stamping mechanism of the ICMP proto-
col (57). However, they did not evaluate the inaccuracy in ICMP timestamping mechanism and they have assumed both sender and receivers are synchronized.

Recently, several studies have explored other forms of delay models (23; 24; 58). The accuracy complexity trade-off is the motivation for all these researches. Duffield et al. (24) have described a varying bin size model for estimating the link delay distribution where the delay bin size is a composition of fixed bin size models. The idea is that the smaller bins are used to capture the small delay values. The larger bins are used to prevent explosion of the numbers of parameters and to capture the delays experienced by slower links. The authors then relate the varying bin size model to the fixed bin size model where the analysis takes place. The construction of varying bin size is chosen a priori or based on the measurements.

In a recent paper by Shih and Hero (23), a finite mixture model is proposed to estimate the link delay probability distribution functions. They model the delay with continuous Gaussian mixture components and assume that the components in the link delay distribution have distinct means and variances.

3.7 Summary

We introduce a new nonparametric methodology for network delay tomography based on unicast end-to-end measurement. Our approach takes advantage of the correlation between the delay experienced by back-to-back packet pairs. We pose the network tomography problem as a maximum penalized likelihood estimation, and develop a fast Fourier Transform based EM algorithm for computing our estimates. The complexity is reduced to
$O(MN^2 \log N)$, where $M$ is the number of links in the tree and $N$ is the number of probes.

We demonstrated the accuracy of the estimation procedure using network-level simulator ns-2.
Chapter 4

Network Radar: Round Trip Time based Network Tomography

In this chapter, we describe the Round Trip Time (RTT) based Network Tomography approach and the challenges associated with its practical use. We investigate the conditions under which RTT tomography is effective by careful designing different laboratory-based emulation experiments.

The remainder of the chapter is structured as follows. In Section 4.1 we describe basic network tomography concepts, challenges in the Internet, RTT tomography and the measurement framework for RTT tomography. Section 4.2 is a detailed treatment of our tomographic inference method including confidence estimates. In Section 4.3 we describe the details of the laboratory and Internet environments, the tests conducted, the operating conditions, the test results and their implications. In Section 4.4, we discuss possible extensions to our current technique. In Section 4.5 we provide further details on studies related to this work. In Section 4.6, we conclude and discuss our future research direction.

4.1 Measurement Framework

The basic idea of unicast tomography is to use back-to-back packet pair measurements, where two closely time-spaced (back-to-back) probe packets are sent from a single sender
to two different receivers. If one were to trace the paths of these probe packets from sender to receiver, they would form a tree with the root at the sender, a common trunk and the leaves at the receivers. We call this RTT-based approach *Network Radar* since it is analogous to the idea of standard radar which sends signals into a medium, collects the "echo" and compares signal to echo strength ratio to estimate the distance to the objects. The two probe packets are assumed to experience nearly the same delay/loss conditions on the shared links. If this is true, then any differences in measured delay/loss are caused by the conditions on the unshared links. By repeated probing to different pairs of receivers, it is possible to construct the (logical) link delay/loss distributions on all branches connecting the sender to the receivers.

### 4.1.1 Challenges

Despite the recent success in the theoretical network tomography developments, the deployment of these techniques in the Internet is problematic. A significant limitation of most methods described in prior network tomography studies is the requirement of *one way* probe traffic measurements and as such, they require *cooperation between sending and receiving hosts*. The only exceptions to this requirement are methods based on passively monitoring losses via acknowledgements (2; 6; 59) These methods are also problematic since losses are extremely rare events in most networks today. These issues limit both the scope of the paths over which the measurements can be made and wide-spread used of tomographic tools. The other limitation is the requirement to have clock synchronizations
between the end hosts. Clock calibration in one way delay measurements have shown to be problematic in Internet measurement (41). By removing the receiver ends cooperation, we also remove the need to calibrate the clocks at the sender and the receivers.

There are several key challenges associated with the practical use of Network Radar. First, like prior tomographic tools, RTT tomography requires route stability, identical behavior on the shared segment, and spatio-temporal independence on unshared segments. Past studies of traffic and structural behavior such as (60; 61) indicate that while the Internet is a dynamic infrastructure, there are certainly opportunities for the basic tomographic assumptions to hold. Second, since our method relies on response packets generated by end hosts which are not necessarily part of any measurement infrastructure, responses generated at the receivers can add extra, random delays to the RTT measurements. These response generation delays can add a significant noise component that limits the accuracy of our tomographic method. Third, a segment of the return paths will be shared by the response packets from the receivers. This could introduce additional correlations into the RTT measurements that are not caused by the shared outward segment of interest (ideally the return paths are uncorrelated). The objective of our work is to develop an RTT-based tomographic tool, Network Radar, that is robust to the dynamics of Internet traffic and structure, and is therefore well suited for wide use. To that end, we present an investigation of the validity of ideal assumptions as part of this work, and endeavor to determine the spectrum of operating conditions over which our technique is effective.

In the next section, we first describe the basic concepts of network tomography. We
follow this with a contrasting description of Network Radar and details of the measurement methods used in our RTT tomography tool. Our description of tomographic methods assumes a single source transmitting measurement probe packets to two receivers. We also assume that the topology is fixed throughout the measurement period (i.e., the routing table does not change and no load balancing* is employed) forming a tree with the source at the root and the receivers at the leaves as depicted in Figure 4.1. The branching node between the source and receivers represents an internal router. Connections between the source, router, and receivers are called segments or logical links. Each segment between may be a direct connection, or there may be “hidden” routers or switches (where no branching occurs) along the path that are not explicitly shown in Figure 4.1. We focus specifically on delay variance estimation on shared segment of the path. This focus is for the sake of working on a concrete problem only and there are nothing inherent in our descriptions that prevent the tomography methods from being extended to loss rate measurements.

4.1.2 Tomography Basics

Basic tomographic measurement and inference ideas are straightforward. They begin by assuming a cooperative measurement environment with time synchronized senders and receivers capable of making one way delay measurements. They further assume that individual link delays along the end-to-end path are independent and stationary. A sender transmits two closely time-spaced (back-to-back) probe packets to two different receivers.

*Load balancing is typically prefix based in the current Internet, as contrast to load based, to avoid packet reordering. Thus the packets follow the same path between a source-destination pair.
Figure 4.1: One way tomographic delay variance estimation in a standard one sender (0) two-receiver (1, 2) network. Variances on shared ($\sigma^2_{\text{shared},s}$) and unshared segments ($\sigma^2_{\text{shared},1}$, $\sigma^2_{\text{shared},2}$) are noted.

The two packets are expected to experience approximately the same delay on the shared segment in their path. The one way delay consists of:

$$y_{\text{one way}} = t_{\text{transmission}} + t_{\text{propagation}} + t_{\text{queuing}}$$

The variances in delays (including a delay equal to $\infty$ which would indicate a lost packet) assumed to be caused primarily by $t_{\text{queuing}}$, and the other terms in the delay can be modeled as a nearly constant quantities.

The delay variance estimation problem is easily understood in the case depicted in Figure 4.1. We index the packet pair measurements by $k = 1, \ldots, N$, and denote the one way delay measurements to be $y' \equiv \{y_1'(k), y_2'(k)\}_{k=1}^N$ where $y_1'(k)$ and $y_2'(k)$ are the $k$th delay measurement to receivers 1 and 2, respectively. We denote the delay on each segment as $d_{\text{shared},i}$, $i \in \{s, 1, 2\}$, then $y_1'(k) = d_{\text{shared},s}(k) + d_{\text{one way},1}(k)$ and $y_2'(k) = d_{\text{shared},s}(k) + d_{\text{one way},2}(k)$. Since probe packets are sent back-to-back, the delay on the shared segment
$d_{\text{outway},b}(k)$ is assumed to be identical in $y'_1(k)$ and $y'_2(k)$. Also note that $d_{\text{outway},i}$, $i = 1, 2$, refers to the time that the probe packets spend traveling from the branching node to the corresponding receiver. Let $\sigma^2_{\text{outway},s}$ denote the delay variance of the shared link and $\sigma^2_{\text{outway},i}$, $i = 1, 2$, denote the delay variances on the unshared paths. Because the delays on the shared and unshared links are assumed to be independent, a straightforward calculation shows that $\sigma^2_{\text{outway},b} = \text{var}(d_{\text{outway},b}) = \text{cov}(y'_1, y'_2)$.

4.1.3 Round Trip Time Tomography

Figure 4.2: Round trip time tomographic delay variance estimation in a standard one sender (0) two-receiver (1, 2) network. Variances on shared ($\sigma^2_{\text{RTT},s}$) and unshared segments ($\sigma^2_{\text{RTT},1}$, $\sigma^2_{\text{RTT},2}$) are noted.

Tomography based on round trip time measurements is a natural extension of the basic ideas explained above. As depicted in Figure 4.2 the principle difference is in the path that the probe packets follow after the branch point. In this case, round trip time delay consists of:

$$y_{\text{RTT}} = t_{\text{one way forward}} + t_{\text{one way return}} + t_{\text{receiver processing}}$$
or equivalently,

\[ y_{RTT} = t_{\text{shared segment}} + t_{\text{unshared segments}} + t_{\text{receiver processing}} \]

The basic assumption that delay variances are caused by queuing delays remain. Besides, the remaining terms in the equation are assumed to be independent. We also assume that the receiver processing delay can be modeled as constant and it ideally zero.

There is almost no change in the formulation of the inference expression for tomography based on RTT packet pair measurements. Again, packet pair measurements are \( k = 1, \ldots, N \) but now round trip time measurements are \( y = \{y_1(k), y_2(k)\}_{k=1}^N \) where \( y_1(k) \) and \( y_2(k) \) are the \( k \)th RTT measurements to/from receiver 1 and 2, respectively, i.e.,

\[ y_1(k) = d_{RTT,s}(k) + d_{RTT,1}(k) \text{ and } y_2(k) = d_{RTT,s}(k) + d'_{RTT,2}(k). \]

Since probe packets are sent back-to-back as in the one-way delay, the delay on the shared segment \( d_{RTT,s}(k) \) is assumed to be identical in \( y_1(k) \) and \( y_2(k) \). Moreover, \( d_{\text{one-way},s} \equiv d_{RTT,s} \), the delay on the shared segment is the same as in the one-way case. We simplify the notation by \( d_s \). The major difference is that \( d_{RTT,i}, i = 1, 2 \) refers to the time that the probe packets spend traveling from the branching node, thru the corresponding receiver and back to the sender. We use the notations \( \sigma^2_s \) to denote the delay variance of the shared segment and \( \sigma^2_{RTT,i}, i = 1, 2 \), denote the delay variances on the unshared segments from the branch point to the receiver and back to the sender. Once again, since the delays on the shared and unshared links are assumed to be independent we can calculate

\[ \sigma^2_s = \text{var}(d_s) = \text{cov}(y_1, y_2). \]

The basic idea is trivial, however, implementing the Round Trip Time approach in the
Internet is challenging. One of the challenges will be to verify if the $t_{\text{receiver processing}}$ can be modeled as constant. Other challenges include assessing packet correlation in return path and assessing our other assumptions such as route stability, and identical behavior on the shared segment.

4.1.4 Measurement Methodology

Our method for taking round trip time measurements is based on sending TCP SYN packets to the Hyper Text Transfer Protocol (HTTP) service (port 80) on a target end host. Any remote host running this service will respond with a SYN-ACK packet. Round trip time measurements can then be made at the sender using simple time differencing between transmission of the SYN and receipt of the SYN-ACK. There are a number of obvious reasons for basing our measurement tool on TCP/HTTP. They include the fact that TCP packets typically receive high priority in routers (unlike ICMP packets), HTTP services are widely enabled on end hosts, and there is typically little delay in generation of SYN-ACK response packets. All of these issues lead to the possibility of Network Radar being used over many paths in the Internet which is one of our objectives.

One of the strengths of the RTT-based approach is that it eliminates the need for careful time synchronization between sending and receiving hosts which was required in prior tomographic methods. In (41), Paxson has shown the difficulty in synchronizing clocks across wide-area network. He showed that this inaccuracy persists even in clocks synchronized by network time protocol such as NTP. Our Round Trip Time approach mitigates
this requirements. However, since delay variance tomography relies on the ability to discern fairly fine grained variations in RTT measurements, the precision of timestamps on SYN and SYN-ACK packets is an important issue. Our measurement method relies on the time-stamping mechanism in the tcpdump utility, which can be commonly found on most systems. In recent work, Paxson outlines some of the problems associated with packet timestamping (28). We compare the results of measurements taken with tcpdump with traces measured by a hardware-based reference timestamping system in Section 4.3.

4.2 Proposed Estimator and Performance Analysis

In this section we present an unbiased estimator for the shared link variance, $\sigma_s^2$, and derive confidence intervals for the estimator. The confidence intervals allow us to automatically detect and reject cases in which the estimator is unreliable.

The standard result of an unbiased estimator for the variance can be found in most statistics textbook. However, in this section, we derive the explicit formula for the unbiased estimator given the conditions and assumptions in our context. The derivation also provides insights in the relationship between the confidence interval and the traffic data.

To begin our analysis, we formally demonstrate that, under the stated assumptions, the covariance of the RTTs is equal to the delay variance on the shared link.

**Proposition 4.1.** Denote the $N$ RTT packet pair measurements by $y \equiv \{y_1(k), y_2(k)\}_{k=1}^N$. The delay on the shared path is denoted by $d_s(k)$, and the delays on the unshared paths are $d_1(k)$ and $d_2(k)$, where $k$ denotes $k$th RTT pair. Assume that the packets in the $k$-th pair
experience an identical delay, \(d_s(k)\), on the shared portion of their paths, and that \(d_s(k)\) is statistically independent of the remainder of the RTT delays, \(d_1(k) = y_1(k) - d_s(k)\) and \(d_2(k) = y_2(k) - d_s(k)\). Furthermore, assume that the delays in different measurements (different \(k\)) are statistically independent. Then \(\text{cov}(y_1, y_2) = \sigma_s^2\).

**Proof of Proposition 4.1** Let \(\mu_i, i = 1, 2\), denote the (unknown) mean RTT in each case and \(m_i, i = \{1, 2, s\}\), denote mean delay in each link. Then,

\[
\text{cov}(y_1, y_2) = E[(y_1(k) - \mu_1)(y_2(k) - \mu_2)]
\]
\[
= E[(d_s(k) - m_s + d_1(k) - m_1)(d_s(k) - m_s + d_2(k) - m_2)]
\]
\[
= E[(d_s(k) - m_s)^2]
\]
\[
= \sigma_s^2
\]

The assumptions of Proposition 4.1 are approximately met in actual practice. The first assumption is that the two back-to-back packets experience the same delay on the shared path. This is reasonable due to the back-to-back nature of the probes, and we experimentally verify this assumption later in the chapter. The second assumption regarding the statistical independence of the delays is reasonable because the cross-traffic tends to be independent on the shared and unshared paths, and by sufficiently spacing the probes in time, the delays in different measurements are fairly independent. Both these assumptions have been verified by experimental work as well. Also, the assumption of independence could
be weakened to only assume that the delays are uncorrelated.

We now state our estimator of the delay variance on the shared path, and verify that it is an unbiased estimator of \( \sigma^2_a \) in the following proposition.

**Definition 4.2.** Denote the \( N \) RTT packet pair measurements by \( y \equiv \{y_1(k), y_2(k)\}_{k=1}^N \).

The RTT covariance estimate is defined as

\[
\hat{\sigma}_a^2 \equiv \frac{1}{N-1} \sum_{k=1}^N (y_1(k) - \bar{y}_1)(y_2(k) - \bar{y}_2)
\]  

(4.1)

where \( \bar{y}_i \) is the sample mean of \( \{y_i(k)\}_{k=1}^N \) for \( i = 1, 2 \).

**Proposition 4.3.** Under the assumptions of Proposition 4.1, \( \hat{\sigma}_a^2 \) is an unbiased estimator of \( \sigma^2_a \).

**Proof of Proposition 4.3** To show that \( E[\hat{\sigma}_a^2] = \sigma^2_a \), i.e. \( \hat{\sigma}_a^2 \) is an unbiased estimator of delay variance on the shared link, let's consider expectation of one term in the summation of Proposition 4.1.

\[
E[(y_1(k) - \bar{y}_1)(y_2(k) - \bar{y}_2)]
\]
\[
= E[y_1(k)y_2(k)] - \frac{1}{N} E \left[ \sum_{j=1}^N y_1(j)y_2(k) \right]
\]
\[
- \frac{1}{N} E \left[ \sum_{j=1}^N y_2(j)y_1(k) \right] + \frac{1}{N^2} E \left[ \sum_{j=1}^N \sum_{l=1}^N y_1(j)y_2(l) \right]
\]
If we investigate this expression term by term, expected value of each term is:

\[
E[y_1(k)y_2(j)] = \begin{cases} 
\mu_1\mu_2, & \text{if } j \neq k \\
\sigma_s^2 + \mu_1\mu_2, & \text{if } j = k 
\end{cases}
\]

Here we use the fact that given a pair of RTT’s, i.e. \(y_1(k)\) and \(y_2(j)\), the elements of the pair are independent if \(j \neq k\), since two packets are not back to back.

Substituting this expression above, expectation of each term is

\[
E[(y_1(k) - \bar{y}_1)(y_2(k) - \bar{y}_2)] = \frac{N - 1}{N}\sigma_s^2
\]

Exploiting the fact that, \(y_i\)'s are identically distributed, expectation of the sum can be written as;

\[
E[\hat{\sigma}_s^2] = \frac{1}{N - 1} \sum_{k=1}^{N} E[(y_1(k) - \bar{y}_1)(y_2(k) - \bar{y}_2)] = \frac{N}{N - 1} \left(\frac{N - 1}{N}\sigma_s^2\right) = \sigma_s^2
\]

Next we investigate the reliability of the estimator \(\hat{\sigma}_s^2\) via confidence intervals. Specifically, we first determine the theoretical variance of the estimator, and then propose an unbiased estimator for this variance. The square-root of the variance equals the standard deviation \(\delta\) of \(\hat{\sigma}_s^2\), which provides confidence intervals of the form \(\hat{\sigma}_s^2 \pm \alpha\delta\), where \(\alpha > 0\).
is the confidence level (e.g., $\alpha = 3$ produces a confidence interval of three standard deviations).

**Proposition 4.4.** Under the assumptions of Proposition 4.1

\[
\delta^2 \equiv E[(\sigma^2_k - \hat{\sigma}^2_k)^2] \quad (4.3)
\]

\[
= \frac{1}{N} E[\hat{y}_1(k)\hat{y}_2(k)] - \frac{N - 2}{N(N - 1)} \sigma^2 + \frac{1}{N(N - 1)} \sigma^1 \sigma^2
\]

where $\sigma_i^2 = \text{var}(y_i)$ and $\hat{y}_i(k) = y_i(k) - E[y_i(k)]$.

**Proof of Proposition 4.4** Since $E[\hat{\sigma}^2_k] = \sigma^2_k$ (according to Proposition 4.3) we have

\[
E[(\sigma^2 - \hat{\sigma}^2)^2] = E[(\hat{\sigma}^2_k)^2] - \sigma^4
\]

and

\[
E \left[ (\hat{\sigma}^2_k)^2 \right] = E \left[ \left( \frac{1}{N - 1} \sum_{i=1}^{N} (y_i(k) - \bar{y}_1)(y_i(k) - \bar{y}_2) \right)^2 \right]
\]

(4.4)

Next, substituting $\bar{y}_i = \frac{1}{N} \sum_{k=1}^{N} y_i(k)$, $y_i(k) = \bar{y}_i(k) + E[y_i(k)]$, and after a bit of algebraic manipulation we have

\[
E[(\hat{\sigma}^2_k)^2] = \frac{1}{N} E[\hat{y}_1 \hat{y}_2] + \frac{N^2 - 2N + 2}{N(N - 1)} (E[\hat{y}_1 \hat{y}_2])^2 + \frac{1}{N(N - 1)} E[\hat{y}_1^2] E[\hat{y}_2^2]
\]
Using the facts that $E[(\sigma_2^2 - \hat{\sigma}_2^2)^2] = E[\hat{\sigma}_2^2] - \sigma_2^4$, $(E[\hat{y}_1\hat{y}_2])^2 = \sigma_2^4$ and $E[\hat{y}_i^2(k)] = \sigma_i^2$ we arrive at

\[
\delta^2 = \frac{1}{N} E[\hat{y}_1^2(k)\hat{y}_2^2(k)] - \frac{N - 2}{N(N - 1)} \sigma_2^4 + \frac{1}{N(N - 1)} \sigma_1^2 \sigma_2^2
\]

Note that the standard deviation $\delta$ depends on the the delay variances on both the shared and unshared portion of the links, as well as the fourth-order central moment on the shared portion (implicit in the cross-moment $E[\hat{y}_1^2(k)\hat{y}_2^2(k)]$). This demonstrates that the unshared portions of the path effectively contribute a “noise” to our measurements; the more variable the delays on the unshared portions, the worse our estimator performs. In practice, we do not have the theoretical value of the standard deviation $\delta$, but we can estimate it from the data as shown in the next proposition.

**Proposition 4.5.** Define

\[
\hat{\delta}^2 \equiv \frac{1}{(N - 3)(N - 2)} \sum_{k=1}^{N} (y_1(k) - \hat{y}_1)^2 (y_2(k) - \hat{y}_2)^2 - \frac{N + 1}{N(N - 3)} (\hat{\sigma}_2^2)^2 - \frac{N - 1}{N(N - 2)(N - 3)} \hat{\sigma}_1^2 \hat{\sigma}_2^2
\]  \hspace{1cm} (4.5)

where $\hat{\sigma}_i^2$ is the empirical variance of the RTTs to destination $i$, for $i = 1, 2$. Then under the assumptions of Proposition 4.1, $E[\hat{\delta}^2] = \delta^2$; i.e., $\hat{\delta}$ is an unbiased estimate of the
theoretical value $\delta^2$.

**Proof of Proposition 4.5** The estimate $\hat{\delta}^2$ has three terms. The expectation of each term is a linear combination of the three terms in the formula for $\delta^2$ in Proposition 4.3. Furthermore, the sum of these linear combinations is equal to the sum of terms in the expression for $\delta^2$, and the result follows immediately from this observation. This is straightforward to verify, but involves a bit of rather tedious algebra.

Finally, we apply the confidence intervals to automatically detect and reject cases in which the estimator $\hat{\sigma}^2$ is unreliable. From Propositions 2 and 4, and the measured data, we can compute $\hat{\sigma}^2$ and a confidence interval of the $\hat{\sigma}^2 \pm \alpha \hat{\delta}$, where $\hat{\delta} \equiv (\hat{\delta}^2)^{1/2}$. If $\hat{\delta} \gg \hat{\sigma}^2$, then the confidence interval is quite large relative to the center point. In such cases, we can be reasonably conclude that the estimator is unreliable. The unreliability could be due to excessive "noise" on the unshared portions (as mentioned above) or large deviations from the assumptions of our theory. Thus, for a given confidence level $\alpha$, we say that the estimator is $\alpha$-unreliable if

$$\frac{\hat{\sigma}^2}{\hat{\delta}} < \alpha.$$ 

In practice, we recommend a level of $\alpha \geq 3$. 
4.3 Experimental Evaluation

We evaluate the capabilities and robustness of RTT tomography in a series of lab-based and Internet-based experiments. The lab-based experiments enable control and instrumentation of all aspects of the test environment. This allows us to verify the capabilities and limitations of the tool in a realistic but limited infrastructure. We also perform Internet experiments to assess the tool’s performance in the wide area network. Its capability in identifying the logical network topology provides an indirect but verifiable (via traceroute) means of assessing the tool’s performance.

Figure 4.3: The laboratory network configuration includes 5 routers and 9 PCs. The sending host is 0 and the receivers are 1 and 2 (logical topology in gray). The boxes xT denote cross-traffic generators and the balls R denote CISCO 7200/7500/12000 series routers. S1 and S2 denote measurement systems used to validate the performance of the RTT based tool and DAG denotes the DAG measurement system placement.
4.3.1 Timestamping Mechanism

The accuracy of the measurements depends on the calibration in time measurements and the timestamping mechanism. In (41), it was pointed out that the important of clock calibration in delay measurement. The author also pointed out that the major cause of uncertainties in these measurements are caused by the clock offset, accuracy, skew and drift. The RTT tomography removes the need to have synchronized clocks at the sender and the receivers. The clock offset, difference between the actual time and the reported time, can be ignored. We are only interested in the relative time, the time difference between the packet is sent and acknowledged, only at the sender. The accuracy, or the frequency of the clock will be studied in more detail in the emulation experiments. In our experience, the clock skew and drift are insignificant compared to the duration of each experiment. The measurement duration is typically less than 2 minutes, while the effect of clock skew is problematic for much longer time scale. The clock drift or clock adjustment might be an issue. However, in our experience, it would affect at most one packet in our measurements. This can be accommodated in our data-processing stage, described next.

The RTT tomography relies on timestamping probe packets when they depart from the sender (SYN packets) and when the responses (SYN-ACK packets) arrives at the sender. The time difference between the SYN and SYN-ACK provides the RTT delay measurements. An accurate time-stamping mechanism is essential and critical to the delay measurements. The time-stamping mechanism used at the sender is the `tcpdump` (62) utility,
a network packet capture tool that can be commonly found on most systems. The exact mechanism as well as the precision of `tcpdump` varies on from system to system. On the Linux systems, the precision of timestamping is 1\(\mu\)sec (6 decimal places) but the actual accuracy depends primarily on the underlying hardware. For the P4-based systems used in our experiments, accuracy on the order of single milliseconds can be expected. We discuss and compare the accuracy of timestamps provided by `tcpdump` with timestamps provided by a hardware-based reference system that uses Endace DAG cards (63) for packet capture. Through our experiments, we conclude that the measurements provided by `tcpdump` do not add significant “noise” and are thus suitable for RTT tomography (on at least our hardware platform) as shown in Fig. 4.6.

4.3.2 Emulation Experiment

Our lab-based experimental environment includes 5 Cisco commercial routers (7200/7500/12000 series) and 9 PCs running Redhat Linux. The bandwidth on all connections is 1Gb/s. The setup is illustrated in Fig. 4.3. Boxes 0, 1 and 2 denote the nodes of interests as in Fig. 4.2. Box 0 refers to the sender and box 1 and 2 are the receivers. Variable background (non-probe) traffic in this environment is generated using *Harpoon* (64), a flow level traffic generator that runs on the systems denoted by \(xT\). Propagation delays on individual links are emulated using a simple configuration of the Click modular router (65). During each experiment, background traffic loads are generated based on input distributions derived from NetFlow logs captured at the border router of University of
Wisconsin. Emulated propagation delays on each link are fixed and remain constant.

Each measurement period consists of 1000 packet pairs sent from node 0 (the sender) to receiver nodes 1 and 2. The send rate is fixed at a rate of 10 probes/sec (100 ms intervals). Practically speaking, this is a low probe rate which should not cause concern for medium to large web sites in the Internet. At the end of each measurement period, we collect packet traces from tcpdump which is running on the sender (node 0) and at two monitoring devices (S1 and S2) along the path to the receivers. The monitors, which of course are not be possible outside of the lab, allow us to verify the performance of our tool by providing ground truth measurements of packet delays. The first monitor, S1, records the back-to-back packet spacing entering the branching router. The second monitor, S2, records outgoing packets from the branching router 2 with extra cross traffic and it provides us the “true” delay variance on the shared link of the path. We synchronize the clocks on the monitoring hosts via network time protocol. The clocks are disciplined from a local stratum 1 time source giving us synchronization on the order of single milliseconds. This is important for verification of one-way delays in our experiments. In order to validate the tcpdump results, DAG-based measurement system is also used.

Data Processing

In this section, we describe the data collection and processing stage as shown in Figure 4.4. in the lab-based experiments. The same procedure is used for the Internet-based experiments. We collect tcpdump data with filtering option “tcp port 80” at the monitoring
hosts during the measurement period. In other words, we are interested in those http probe packets only. In the probe identification stage, we identify the probe packets by matching the source and destination addresses and locate the probe pairs from the sequence numbers. Our analysis does not consider packet pairs in which one or both packets are retransmitted or dropped along the forward or return paths. The retransmitted packets are dropped from our measurements in our filtering stage. The basic idea is to ignore packets whose measured round trip time is larger than twice the median RTT on each path. In other words, we first compute the median RTT to node 1, filter packets from the sequence and retain only those within the bounds of the threshold. Then we carry out the same procedure for
Table 4.1: An example of the lab-based experiments. The average RTT to node \( i = 1, 2 \) is \( \bar{y}_i \). The standard deviation of RTT to node \( i = 1, 2 \) is \( \text{std}(y_i) \). The SNR is given as the ratio between the estimated shared link delay variance \( \hat{\sigma}_x^2 \) and its confidence interval \( \sqrt{\hat{\delta}^2} \).

<table>
<thead>
<tr>
<th>Case</th>
<th>( \bar{y}_1 (ms) )</th>
<th>( \bar{y}_2 (ms) )</th>
<th>std(( y_1 ))(ms)</th>
<th>std(( y_2 ))(ms)</th>
<th>( \hat{\sigma}_x^2 (ns^2) )</th>
<th>( \sqrt{\hat{\delta}^2} (ns^2) )</th>
<th>SNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1.4</td>
<td>12.2</td>
<td>0.246</td>
<td>0.236</td>
<td>45.50</td>
<td>3.205</td>
<td>14.20</td>
</tr>
<tr>
<td>i</td>
<td>1.3</td>
<td>12.1</td>
<td>0.292</td>
<td>0.278</td>
<td>57.39</td>
<td>2.833</td>
<td>20.26</td>
</tr>
<tr>
<td>ii</td>
<td>1.5</td>
<td>12.4</td>
<td>0.063</td>
<td>0.047</td>
<td>0.489</td>
<td>0.151</td>
<td>3.23</td>
</tr>
<tr>
<td>iii</td>
<td>1.5</td>
<td>12.2</td>
<td>0.131</td>
<td>0.029</td>
<td>0.778</td>
<td>0.207</td>
<td>3.75</td>
</tr>
</tbody>
</table>

those destined to node 2. These packets are filtered according to the median RTT computed from sequence destined to node 2. It is important to ensure if we remove a packet from one sequence, we have to remove its pairs from the other sequence. We then compute sample variance and the confidence level from the remaining data.

An example of lab-based experiment is shown in Table 4.1. The average RTTs \( y_i \) measured from node \( i = 1, 2 \) are shown in the first two columns. The variability of these RTTs to each destination are shown in the third and forth column. The fifth column shows the estimates of the variance computed from the covariance of \( y_1 \) and \( y_2 \). The sixth column provides the confidence interval given by Equation 4.5. The last column computes the Signal to Noise Ratio (SNR) defined to be the ratio between the estimator and its confidence.

The confidence interval provides a measure to the reliability of the data. We have studied three different cases in the experiments in Table 4.1. In case (i), we have moderate traffic on all links; (ii) we have low traffic on shared link and moderate traffic on unshared links; and (iii) different load on each unshared path. The confidence interval is able to capture all these scenario. The confidence is low when there is lack of correlation in the
packets because there is insufficient traffic on the shared link, illustrated in case (ii). Confidence is low when the variability on the each unshared path differs significantly, comparing column 3 and 4 in case (iii). By focusing at the SNR, we notice that when SNR falls below 3, the measurements are not reliable. This typically results from a lack of correlations within the data and the large variability in the unshared portion of the path(s) (differ by order of magnitude), as shown in Table 4.1. In the remainder of the chapter, we focus only on those measurements with SNR > 3.

In Section 4.3.2, we described a simple filtering scheme in our data collection process to assess the data fidelity. There are other signal processing techniques or filtering schemes which might improve the estimates. One example is the wavelet smoothing technique described in (66). There, the authors apply an adaptive soft threshold to wavelet coefficients based on the variance of the data (noise level) and the number of measurements available. The authors conjectured that links with light traffic have a noise-like structure and those with heavy traffic shows pulses with larger magnitude. Instead of implementing a low-pass filter to capture the dominant components at low frequencies, they employ the adaptive nature of wavelet denoising to smooth the signal, thus removing the light traffic and increasing the correlation coefficient between two flows traversing a congested link. Unfortunately, it is not clear how many vanishing moments should be chosen in our framework. In this work, we are not trying to improve our estimates but to provide a useful and robust tool which allows the tomography techniques to be widely used.
Results

Figure 4.5: Plot of standard deviation from direct measurement (horizontal axis) vs. estimated delay standard deviation using RTT-tomography tool (vertical axis).

Figure 4.5 depicts the accuracy of our tool by comparing the estimates to the "true" value of the shared path delay standard deviation. The "true" value ($\sigma_s$) for the one way delay on the shared path is the measured time difference of TCP SYN packets at the sender and at the second monitor $S2$ in Figure 4.3. The estimates are computed directly from the measurements $y_1$ and $y_2$. A moderate level of background traffic was used during this experiment and no packet loss was observed. The one-way delay between node 0 and node 1 was fixed at 0.6ms while the node 0 to node 2 delay was fixed at 6ms. Ideally, the estimates should be identical to the "true" value and fall onto the $45^\circ$ line. The discrepancy might arise if the timestamping mechanism is inaccurate or unreliable, or if the back-to-back assumptions described in the previous section are violated. Nonetheless, the estimates are close to the real value.
Operating Condition Study

One of the primary objectives in our lab-based experiments is to understand the operating conditions under which RTT tomography is practical. To investigate potential sources of errors that would make RTT tomography ineffective, we examine the effects of; (1) the reliability of the timestamping mechanism (comparing `tcpdump` and others), (2) varying levels of background traffic, (3) varying spacing between packet pairs as they are emitted by the sender (back-to-backness), (4) load on receiving end-hosts which can cause variability in generation of response packets (SYN-ACK generation delay), and (5) number of measurements as compared to the resulting estimation error. Except the case where we investigate the effects of the background traffic, we used a fixed moderate background traffic load.

1. **Time-stamping mechanism** We evaluate different mechanisms that can affect timestamp accuracy by experimenting with Realtime Linux, timestamping in kernel, and measuring the CPU cycle counts at the sender.

**Realtime Linux** Real-time Linux guarantees that a process will be to be executed in a time less than the worst-case response time of regular systems. There are several Linux based open source real time systems available. Unfortunately, not all of them supports real time in socket operations. Moreover, the ones that support real time sockets only support UDP packets, not TCP. Therefore, real-time Linux is not useful for our RTT tomography measurements.
**Timestamping in kernel** Timestamping in kernel refers to modifying the kernel timestamping mechanism to improve the accuracy of `tcpdump`. Unfortunately this mechanism is quite complicated and numerous compatibility issues need to be resolved. After inspecting the kernel code `dev.c`, and `3c59x.c`, we note that the timestamping already occurs just before the packets depart from the network interface card and when they are received by the network interface card. However, the accuracy of timestamps is still limited by scheduler accuracy to the scheduler.

**CPU cycle counts** Most system clocks suffer from clock skews and drifts. Pásztor and Veitch (39) proposed to improve the clock accuracy by counting the CPU cycles. This is accomplished by replacing calls to the system function “gettimeofday” with a special purpose function that directly reads the hardware register which stores cycle counts. The cycle counts are converted back to time units using an estimate of the processor clock frequency obtained using the Network Time Protocol (NTP). This is shown to improve both clock and timestamping accuracy. We did not pursue this approach for this study since it affects the scheduling of packets: the average packet spacing with this approach was 500ms, which is relatively large for our RTT measurements. That said, the general approach is promising and we plan to investigate it further in our future work.

**DAG card** The Endace DAG cards (63) is the current state of art in hardware timestamping. It provides timestamping with high accuracy, with no packet loss and no
extra delay. Most network interface cards suffer from corrupted, delayed or lost packets when the traffic volume is high. We use systems with DAG cards to assess how \texttt{tcpdump} timestamping variability at the sender PC effects RTT measurements. During these experiments we capture the packet departure and arrival times using both \texttt{tcpdump} and systems with DAG cards. By comparing the results in Fig. 4.6, we conclude that the variability in RTT measurements caused by \texttt{tcpdump} does not have a significant impact.

In the rest of the chapter, the delay measurements are computed from \texttt{tcpdump} timestamping mechanism.

2. **Background Traffic** With respect to background traffic, the capabilities of RTT tomography are proportionally dependent on the load on the shared segment and inversely proportional to the load on the unshared segments. Specifically, the estimator
Figure 4.7: Significant of background traffic in the estimates. High SNR symbolizes high accuracy in the estimates. The signal to noise ratio (SNR) increases when the background traffic (correlation) on the shared segment increases. Unshared segments had moderate load in these experiments.

does not work when there is no load on the shared segment and improves as the load on the shared segment increases. This is depicted in Figure 4.7 which shows signal quality improving as load on the shared segment increases. If there is some load on the shared segment but no load on the unshared segments, the estimator is unaffected. However, as load increases on the unshared segment the capabilities of the estimator degrade somewhat. The fact that response packets share the final link on the return path does not affect the performance of the estimator as long as they do not wind up in the same queue at the same time, and ending up in the same queue at the same time on the return path is highly unlikely.

Note that the accuracy of our estimator not only depends on the variance on the shared segment, but also the variances on the unshared segments. As these variabilities increase, our confidence weakens, as shown in Equation 4.5.
3. **Back-to-back assumptions** Figure 4.8 depicts the accuracy of the results by varying the packet spacing within the probe pair while the spacing across pairs remains unchanged. The results agree with our expectation that the accuracy of the estimates decreases as the back-to-back assumption weakens. When the packets are not well correlated, the packets no longer have similar experience on the shared link. In our experiments, with exhaustive trials, the smallest achievable packet spacing is $3\mu s$. This depends on the scheduling of the packets. We notice that by scheduling more than 2 packets at the same time, the spacing between probes reduces. We place an ICMP echo reply packet destined to the nearest router prior sending the two probes, we achieve the smallest spacing. However, as we increase the number of extra packets prior sending the two probes, there is no improvement in the effort in reducing the spacing. If no special scheduling is performed, the average spacing between the
Figure 4.9: Plot of delay standard deviation on the shared link measured from sender to the branching node destined to receiver 1 against that to receiver 2.

packets is \(100\,ms\).

In our experience, the effectiveness of the tool depends on how the connections are scheduled. When more connections are initiated at roughly the same time, the spacing between packets that are placed on the network interface card shortens. Nonetheless, if too many packets have too be generated to ensure the back-to-backness, we are congesting our outgoing link with our own packets and are then potentially biasing our measurement results. Also, spacing the probes further apart to ensure the independence assumption lengthens the measurement duration for each test. This gives more opportunity for conditions to change (e.g., abrupt average RTT change as a result of routing table updates) during a test which, again, can adversely affect results. We assume the packets are correlated on shared portion of the path, thus the shared path delay variance directly measured by the packets destined to each receiver should be identical. We compare the delay variance computed by the two measure-
Table 4.2: Average SYN-ACK generation delay and its variance with varying end host load

<table>
<thead>
<tr>
<th>HTTP Connections</th>
<th>Average Delay</th>
<th>Delay Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.23μs</td>
<td>$1.8 \times 10^{-12}s^2$</td>
</tr>
<tr>
<td>201</td>
<td>28μs</td>
<td>$3.4 \times 10^{-11}s^2$</td>
</tr>
</tbody>
</table>

ments, $y_1$ and $y_2$. Figure 4.9 illustrates the correlation (back-to-backness) of the probe packets at the branching routers. As the measurements from each packet in a pair form a 45° line, and thus highly correlated, this indicates that our back-to-back assumption holds even in the presence of traffic.

4. **End host load** Delays in the generation of SYN-ACK response packets at the end hosts depends on a variety of factors including scheduling and the load. To evaluate these effects, we adjust the end host load by varying the number of active http connections to the Apache 2.0.52 http server on that system. The results of average SYN-ACK generation delay are illustrated in Table 4.2. The variability of the SYN-ACK generation delay is negligible even for large number of active simultaneous connections, e.g., 201 connections as shown in Table 4.2.

5. **Number of measurements** In the final set of lab-based experiments, we evaluate how the number of packet pair probe measurements affect delay variance estimation. We conduct experiments using 100, 200, 500 and 1000 packet pair probes. As expected, the confidence of the estimator decreases with decreasing number of measurements. The confidence of the estimates is given as the standard deviation of the estimator and the signal to noise ratio (SNR) is defined in our case to be the ratio
Figure 4.10: Plot of signal to noise ratio (vertical axis) given different number of packet pair probe measurements.

between the estimator and the confidence. We compute the SNR for each measurement sets (varying the number of probes), the SNR increases with the number of probes, as shown in Fig. 4.10. Typically, the number of probes needed depends on the variability in traffic and the back-to-back assumption.

One possibility is to approximate the number of probes required according to Equation 4.5 from the accuracy required and the variability in traffic. The number of measurements required for a specific accuracy level can be bounded by Equation 4.5. The equation provides an explicit relationship between $N$ and $\hat{\sigma}^2$. We can solve for $N$ given a confidence level acquired. By equating Equation 4.5, we have a $3^{rd}$ order polynomial of $N$,

$$\hat{\sigma}^2 N^3 + (B - 5\hat{\sigma}^2) N^2 + (6\hat{\sigma}^2 - A - B - C) N - (2B + C) = 0$$
where \( A = \sum_{k=1}^{N} (y_1(k) - \bar{y}_1)^2 (y_2(k) - \bar{y}_2)^2 \)
\( B = (\hat{\sigma}_y^2)^2 \)
\( C = \frac{\gamma_1 \gamma_2}{\sigma_1^2 \sigma_2} \)

Assuming all the quantities (A, B and C) are known, we could solve for \( N \) directly or through numerical analysis. Unfortunately, these quantities are not known \textit{a priori}, one would have to apply a two-step procedure: (1) estimate \( A, B \) and \( C \) using a small number of measurements, (2) using the estimate and the acquired accuracy \( \hat{\delta}^2 \) to solve for the number of measurements \( N \) required.

Theoretically, one can use infinite number of measurements to achieve high accuracy. However, in practice, the measurement period increases as we increase the number of probes. The measurement period should always be shorter than the routing table updates. If one increases the probe frequency, it is possible to interfere with the normal traffic and induces congestion on the outbound link. The processing load at the receiver ends might also be increased.

4.3.3 Internet Experiments

We carried out RTT tomography tests from a measurement nodes at two universities, University of Wisconsin at Madison (UW) and Rice University (Rice). We randomly selected
20 web servers located at other universities in the continent as our target receivers. All of these universities are members of Internet2 (67) and are connected through Abilene backbone. The Abilene network consists of 11 Point of Presences (PoPs) with OC192 (10 gigabits per second) links and typically are under 20% utilization. All experiments carried out from UW branch at the Abilene PoP Chicago and those from Rice branches at Abilene PoP Houston. Average utilization at PoP Chicago during our experiments was about 15% and utilization at PoP Houston was around 2%. There are 7 hops from both nodes UW and Rice to their respective branching routers. There are on average 15 visible hops between the source and receiver nodes based on counting the number nodes that respond to traceroute. We carried out experiments over a month and collected 4 sets of measurements over 10 pairs of randomly selected receivers from nodes Rice and UW.

An example of results are shown in Table 4.3 and in Table 4.4. We notice that some of the estimates have small confidence interval. Some possible sources of error have been discussed in section 4.3.2. More specifically, it is concluded that given the probes are closely spaced at the sender, and the two round trip time measurements can be differentiated, this ensures that the packets are not correlated on the return path. The main source of error in this case is caused by the background traffic. If we investigate the data given in Table 4.3 and Table 4.4, the SNR is always small (less than 3) when the variances differ by an order greater than 5. This is also discussed in section 4.3.2. The goodness of the estimator also depends on the variances on the unshared path, as they increase our confidence goes down (Equation: 4.5). Moreover, we notice some of the estimates are negative. In other
Table 4.3: An example of the Internet experiments from node UW. The packet pairs are destined to Receiver 1 and 2. Their mean round trip times are $\bar{y}_1$ and $\bar{y}_2$ and their variances are $\sqrt{\sigma_1^2}$ and $\sqrt{\sigma_2^2}$. The estimated shared link variance is shown in $\hat{\sigma}_s^2$ and its confidence (std) in $\sqrt{\hat{\sigma}_s^2}$. The SNR is shown in the last column.

words, there are no correlation between the probes. In these circumstances we conclude there is not enough shared segment traffic, therefore, almost all of the variability is caused by unshared segments which are assumed to be independent. In fact, this is the case of PoP Chicago. We conclude that there is not enough traffic to create any correlation.

We conducted the experiments by sending probes at a rate of 4probes/sec (250ms intervals) to pairs of receiving sites. At the end of each measurement period, we collect tcpdump results at the senders. We compute the delay variance of the shared portion of the links by computing the covariance of traffic destined to pairs of receivers. For each measurement, we also compute the confidence of the estimate, derived at Section 4.2. The experiments have been carried out throughout different times of the day over a period of one month. To our experience, the confidence level remains high when there are high utilization in the Internet (daytime vs. nighttime). However, when losses in probe packets occur, the accuracy of the estimator might also reduce. In case of congestion, the average
<table>
<thead>
<tr>
<th>Receiver ID</th>
<th>$\bar{y}_1$ (10^{-3}s)</th>
<th>$\bar{y}_2$ (10^{-3}s)</th>
<th>$\sqrt{\sigma_1^2}$ (10^{-3}s)</th>
<th>$\sqrt{\sigma_2^2}$ (10^{-3}s)</th>
<th>$\sqrt{\sigma_2^2}$ (10^{-7}s^2)</th>
<th>$\sqrt{\delta^2}$ (10^{-7}s^2)</th>
<th>SNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>USU UIdaho</td>
<td>63.00</td>
<td>76.10</td>
<td>9.00</td>
<td>9.10</td>
<td>777.51</td>
<td>34.77</td>
<td>25.00</td>
</tr>
<tr>
<td>USU OU</td>
<td>63.00</td>
<td>29.20</td>
<td>8.30</td>
<td>5.60</td>
<td>89.78</td>
<td>16.53</td>
<td>5.55</td>
</tr>
<tr>
<td>USU VCU</td>
<td>62.20</td>
<td>43.30</td>
<td>8.90</td>
<td>1.30</td>
<td>10.26</td>
<td>3.48</td>
<td>2.94</td>
</tr>
<tr>
<td>USU VER</td>
<td>71.00</td>
<td>57.20</td>
<td>10.50</td>
<td>1.80</td>
<td>12.63</td>
<td>6.81</td>
<td>1.18</td>
</tr>
<tr>
<td>VER VCU</td>
<td>65.30</td>
<td>54.10</td>
<td>7.50</td>
<td>7.50</td>
<td>415.56</td>
<td>22.65</td>
<td>20.00</td>
</tr>
<tr>
<td>VER UWash</td>
<td>57.90</td>
<td>59.80</td>
<td>2.80</td>
<td>1.90</td>
<td>26.63</td>
<td>6.05</td>
<td>4.34</td>
</tr>
</tbody>
</table>

Table 4.4: An example of the Internet experiments from node Rice. The packet pairs are destined to Receiver 1 and 2. Their mean round trip times are $\bar{y}_1$ and $\bar{y}_2$, and their variances are $\sqrt{\sigma_1^2}$ and $\sqrt{\sigma_2^2}$. The estimated shared link variance is shown in $\sqrt{\sigma_2^2}$ and its confidence (std) in $\sqrt{\delta^2}$. The SNR is shown in the last column.

RTT changes, this violates the stationary assumptions in our setup.

**Topology Identification**

We use topology identification as a *verifiable* method for demonstrating the performance of the RTT tomographic tool. Indeed, one of the significant challenges of network tomography studies in general is *in situ* evaluation since access to network internal instrumentation is so rarely available. In the context of topology discovery, end-to-end topology measurement tools such as traceroute or Scriptroute (68) which can be used to validate the logical topology structures generated using the aforementioned RTT tomography and clustering methods described next.

One major assumption in our framework is that the delays on the shared and unshared links are independent. Therefore, the end-to-end delay variance will be the sum of the link delay variances along the path. One can also use this information to identify the underlying
network topology configuration. This is done by interpreting estimates of delay variation as a measure of the physical extent of the shared path. Let $p_{ij}$ denote the shared path between the source and a pair of receivers $i$ and $j$. Let $i$, $j$ and $k$ be three receivers and suppose that $p_{ij}$ is longer (from a physical distance perspective) than $p_{ik}$ (clearly one of the shared paths has to be included in the other). The delay variance associated with $p_{ij}$ (denoted $\sigma^2_{p_{ij}}$) is going to be larger than the delay variance associated with $p_{ik}$. This partial ordering of the delay variance of the shared paths can be used to identify the underlying topology. For example, refer to Figure 4.11, $\sigma^2_{p_{1,2}}$ will be greater than $\sigma^2_{p_{i,2}}$ for any other receiver $i$, revealing that receivers 1 and 2 are siblings in the logical tree. This property can be exploited to devise a simple and effective bottom-up merging algorithms that identifies the full, logical topology from the pairwise delay variances. This problem is indeed one of hierarchical clustering.

In general we do not have access to the exact delay variances, but only noisy measurements of those. In particular, the variability of the measurements for each pair of receivers
may be significant. In (29) a hierarchical clustering technique that is suitable for the topology identification problem is proposed. This technique is based on a likelihood model capable of accounting for the different statistical properties of the measurements for each pair of receivers. Two clustering techniques building on the likelihood model are proposed in (29): (i) a deterministic, greedy approach, with very low computation complexity; (ii) a Markov Chain Monte Carlo method that attempts to find a globally optimal (in a likelihood sense) topology, at the expense of computation power. In this chapter we apply these techniques to infer the topology solely from RTT covariance measurements.

**Results**

![Diagram](image)

Figure 4.12: Comparison between (a) “true” topology and (b) estimated topology for 6 universities on the Abilene Network. The university homepages indexed from 1 – 6 are Oklahoma University (OU), Virginia Commonwealth University (VCU), Vermont University (UVM), Utah State University (USU), University of Washington (UWash) and University of Idaho (UIdaho). Probes are sent from Rice University.

We collect a set of estimates with good confidence given by the standard deviation of the estimates. If the ratio between the standard deviation of the estimate and the estimate (SNR) is more than 3 (as recommended in Section 4.2), we use them as an input to the
topology identification algorithm. The inputs to the clustering algorithm are the estimates and its variability, as the mean and the variance. We normalize the variance by the number of measurements. The results of topology measurement experiments based on RTT tomography is shown in Figure 4.12. The “true” topology is given by traceroute. The estimated topology is very close to what we expect. The estimated topology results as an artifact of a binary tree.

**Implications**

In our approach, the goal is to enable tomography from a single node, the RTT technique has to be useful given various traffic conditions arise in the Internet. One of the major concern is the correlation in return path. In particular, probes can and almost always do share a segment of the return path. In the Abilene network, the packets indeed share portion of of links not only on the outgoing links but also on the inbound links. The performance of the tool is dependent on the correlation in packets. We have to ensure the packets do not enter the same queue, thus, no correlation between them except on the outbound links up to the branching router. In our experiments, we carefully select those destination in which the average RTT to one receiver is significantly higher than that to the other receiver, to ensure that the packets would not be correlated on the return path. The distance or the number of hops between the sender and each receiver is not important but the average RTT to each of them. In the case when the packets to both receivers have similar average RTT, our tools might have difficulty in differentiating the correlation between the the packets in
both forward and return path. The confidence level might appear random, it can be either be over- or under- estimated depending on the correlation on the shared segment on both forward and return path.

As we have shown in the Internet experiments (as well as those in the laboratory) that the confidence level can provide an automatic measure to the estimates when there is no correlation in the return path. The confidence depends on the variability on the shared and unshared portion of the links. Low confidence level indicates that some assumptions have been violated. High confidence level indicates the delay variance on the shared links can be trusted and are useful for topology identification.

Unlike laboratory experiments, we cannot validate the estimates by comparing to the true value. We rely on the topology identification using those estimates with high confidence interval. The topology identification, however, depends on all confidence measures entries. Those paths with relatively small confidence might not be correctly estimated using the clustering algorithm. However, we can apply different clustering techniques (for example the two described above) and check for consistency in the topology estimated.

The number of measurements used in our Internet experiments can be used as an initial setup. We can adjust the number of measurements to improve the quality of the estimates according the the polynomial relating the accuracy and the number of measurements needed. The accuracy of the estimates depends on the traffic variability on both shared and unshared links. As long as the average RTT between the two receivers are significantly different, our confidence interval can be used to filter out unreliable estimates. Our technique
highly relies on the correlation on the shared portion of the outgoing path, not the time or the frequency which the measurements are carried out.

4.4 Discussion

In this chapter, we described a RTT based tomographic tool, Network Radar, for one-sender-two receivers framework. The tool can be conducted anywhere in the Internet the same way as other network diagnostic tools. The tool is ready to be used and it is in the final stage in integrating the ability to compute the confidence interval. The beta version will be available for download in the near future. We have tested the tool in both controlled environment as well as in the wild Internet. We have shown that the confidence interval derived from our estimator provides a good estimate to the traffic condition. By collecting measurements with high confidence interval, one could use it for topology identification. While it may be obvious, we note that although topology discovery based on network tomography can be useful, it will not provide internal node labels (IP addresses) like other topology measurement tools.

We have detailed some of the operating conditions in which our tool is effective. We summarize them briefly here. The effectiveness of the tool depends on two major components: (1) correlation on shared link and (2) the unshared link variability, given the fixed number of measurements.

1. Sufficient correlation on shared link: The back-to-back assumption requires packets to have correlated experience up to the branching node. The correlation weakens
as the cross traffic enters between the packets within a pair. One can reduce the probability for widening of spacing between the probes by scheduling the packets as close as possible at the sender. However, if there is no or little cross traffic along the shared link, the estimate suffers from lack of correlation between probes, thus the confidence remains low.

2. Unshared link variability: The unshared link variability depends on the traffic level and the system load from the branching router to the receiver and back to the sender. From section 4.3.2, we have pointed out that if there are too much variability on one or both of the unshared links, our estimator will be weakened. From our traceroute experiments, we notice that there are a larger number of hops (physical links) from the branching router to the receiver node than those from the sender to the branching router. The more routers the packets have to traverse, the higher the probability in interfering cross traffic. Moreover, the links away from the core network typically have a lower bandwidth, it is possible to have higher delay variance at these links. Despite the distance (delay) from the sender, as long as the assumptions hold, the confidence interval will provide an indication of the reliability of the results. Regarding the variability introduced by end host traffic loads, we have studied traffic load up to 201 TCP connections. In our study, the SYN-ACK generation delay variance is negligible. We are aware of the fact that it is possible for packets to become correlated on the return path. Despite the asymmetric nature in routing, there is
usually some common outbound and inbound links close to the end-host, e.g. at the
sender host. We recommend not to use receivers with similar average round trip time
in avoiding this correlation. These are some of the components which affects the
confidence of our estimates. Nonetheless, the confident interval provides important
insight in the usefulness of the data.

We have also discussed the number of measurements needed for the tool to become
effective. There are different ways one could distribute the probes. In our setup, we pick a
web-server from each university. We could potentially select a group of web-server from
that university, and then by randomly selecting pairs of web-server, one from each univer-
sity. We could avoid to overload the web-server by distributing the load across different end
hosts. By combining these measurements, we could potentially improve our estimates. The
same idea can be extended to receivers not necessarily in the same domain but share the
links of interest. One can also distribute the probes depending on the variability observed.
Given a fixed number of probes (or size of measurement window), one could distribute the
probe according to their variability using a subset of the probes. We focus on sending probe
to destination with high variability and send a smaller amount of probes to destination with
small variability. The total number of measurements from the sender can thus keep con-
stant while achieving a higher accuracy in the estimates. The selection of destinations will
be studied in our future work. Another interesting extension is to decide the combinations
of receivers when we are interested in a particular segment in the Internet.

Other than the back-to-back probing approach used in this thesis, there are other prob-
ing techniques in proposed in recent tomographic studies, e.g., packet stripes (10), fat-boy sandwich (4) and pathChirp (69). These methods rely on the special design of the packets, for example, the spacing between the packets, the size of the packets and the number of packets. We have tested our RTT based tomographic approach using the fatboy sandwich, where a 1500 byte ICMP packet is placed between two small TCP SYNs. The SYN packets are destined to one receiver, while the ICMP packet is destined to another receiver. The basic idea is that the second TCP SYN packet in the sequence has to queue behind the ICMP packets at all routers along the shared portion of the path between the two receivers. This provides a handle in the average link delay up to the shared portion of the link by computing the difference in the spacing between the first and second SYN packets at the receiver end, refer to (4) for details. In our setup, the shared portion of the path is significantly shorter than the unshared portion. The signal (average delay on shared port) is significantly less than the noise component introduces by the uncertainties on the unshared portion. The time it takes to process the packets at the end-host becomes dependent on the server load. The time measurements become more sensitive. The results using the fatboy sandwich using the RTT tomography approach are not conclusive. The experimental results are not included in this thesis.

In this study, we have focused on delay variance estimates on the shared path from the sender to the two receivers. The delay variance estimate results have naturally extended and resulted in identifying the underlying topology. In most tomographic studies, we either assume the topology is known and infer the link performance in terms of link loss rates
and the delay statistics (delay variance and delay p.m.f.). And when the topology is not known, delay measurements are used to infer the topology. Our tool does not require the knowledge of the topology and topology identification is a nice by-product of the delay variance estimates. Other tomographic interests might involve the joint loss and delay variance estimation. Pure link loss estimation is challenging, the packets can be lost on the forward and return path and it is difficult to provide a confidence measure in this context. We will investigate and study other use of RTT tomographic techniques in our future work.

4.5 Related Work

Network tomography study has been an emerging area in the past few years. Different inference techniques in estimating loss rates, delay statistics and topology identification are proposed (3; 11; 14; 16; 29; 51; 52; 70; 71). Most of these techniques are promising but are not widely applicable because of the requirement to have cooperation from receivers to collect one-way measurement. Some of these techniques also require clock synchronization between end hosts or they depend on special cooperation from the internal routers. The measurement limitations have largely prevent the tomographic techniques to be used widely.

There are other RTT-based measurement studies in inferring path characteristics. Unlike ours, they are based on either the time-to-live (TTL) (14; 52) or the timestamp (70) options in the Internet Control Message Protocol (ICMP). The rising security concerns related to the use of ICMP options have evoked many system administrators to block ICMP
packets or to rate limit them, thus reducing the effectiveness in ICMP measurements. Our methodology, on the other hand, is easy to deploy and is widely available. It is based on the Transport Control Protocol (TCP) three way handshake mechanism, while the majority of the traffic is made of TCP. The same three way handshake mechanism for RTT measurements is used in Sting for one-way loss measurement (72) and Synack for RTT statistic studies (73). However, these tools do not provide localized information.

Bohacek and Rozovskii (59) proposed a Bayesian approach to network tomography based on packet losses and RTTs. Using parametric probability models for queue congestion, their objective is to estimate level of congestion based on the observation of drops. The performance of their method hinges on the occurrences of packet drops (which are extremely infrequent in most networks) and the accuracy of the assumed parametric models. In contrast, our method is based solely on RTTs, not drops, and makes much less restrictive assumptions. Besides, the study was mainly carried out in ns−2 simulation, it is not obvious what are the RTT measurements and the TCP congestion control was not implemented in the ns−2 release at time of publication.

Duffield and Lo Presti (51) have conducted delay variance tomographic study in a multicast setting. They evaluated link delay variance from one-way end-to-end measurements in an analytical framework and in ns−2 simulations. The estimates derived are biased. Their method also assumes the availability of multicast routing, synchronized clocks and the ability to measure at both sender and receivers. In this chapter, our estimator is unbiased and it provides an automatic measure to our estimates. Our approach not only mitigates all
of these requirements, but also is shown to be useful in identifying logical topology in the Internet. The ability of topology identification validates the effectiveness of the tool.

4.6 Summary

Many of the prior methods for network tomography rely on coordinated measurement infrastructures. This has limited both the share of the Internet over which the tools could be used, and the number of people who could potentially take advantage of this useful inference technique. In this chapter we present a network tomography method based on the use of round trip time measurements. While the inference methods used in RTT tomography are not vastly different than prior methods, the important question is how practical is this approach given the dynamic nature of the Internet? We address this question by first developing a delay variance estimation method for round trip time measurements that includes a confidence estimator. The confidence estimator is critical for Network Radar since probe packets are subject to more dynamic conditions than probes in a one-way coordinated infrastructure. The confidence estimator enables inaccurate delay estimates to be rejected.

We conducted an empirical evaluation of Network Radar in both a lab-based environment and in live Internet tests. Both sets of tests were conducted with a packet-pair measurement tool that uses simple TCP SYN/SYN-ACK differencing. The lab-based tests demonstrated that Network Radar is very effective over a range of operating conditions that makes it practical for use in the Internet. The Internet-based tests use Network Radar
to establish end-to-end topology between a sender and ten randomly chosen end hosts. By validating the topologies established via Network Radar with standard traceroute, we again show that our technique is practical.
Chapter 5

Optimal Network Tomography

In this chapter, we formulate network tomography as an optimal probe allocation problem, in which a fixed number of network probes are optimally distributed to minimize the squared estimation error of the tomographic process. The analysis reveals the dominant sources of ill-conditioning and issues scalability of network tomography. We employ resource allocation optimization techniques to design optimal probing schemes, and develop algorithms for optimal probe allocation based on topological considerations and noise characteristics.

The remainder of the chapter is structured in the following manner. In Section 5.1, we describe the measurement infrastructure. In Section 5.2, we describe the problem formulation and the objective function to be optimized. In Section 5.3, we introduce basic concepts of optimal experimental design and focus in A- and D-optimality. In Section 5.4, we analyze the scalability of the probe allocation problem. In Section 5.5 we describe the results of $n \approx 2$ simulations to explore the squared error using optimized probe allocation. In Section 5.6, we describe some of the related works. In Section 5.7, we make some concluding remarks and indicate avenues of future research.
5.1 Measurement Infrastructure

The goal of network tomography is to infer internal link level characteristics from end-to-end measurements. This is commonly done by probing the network with special-purpose packets designed to indirectly elicit information reflecting of the congestion characteristics on internal links. Because probe data is strictly based on end-to-end measurements (e.g., losses, delays), the raw data must be transformed, through a matrix inversion or analogous process, to obtain link-level characteristics.

![Diagram of a tree-structured topology]

Figure 5.1: An example of tree-structured topology with single sender (node 0) and 33 receivers (node 6 through 18). The logical topology is composed of only the solid nodes. The edge links (thinner lines) are slower and the internal links (thicker lines) are faster.

The required inversion is determined by the topology of the network under study. Typical network tomography schemes involve probing several receivers from a single sender. Standard routing protocols thus lead to tree-structured topologies, with the sender at the root and the receivers at the leaves. In most cases, the topology can be found using standard diagnostic tools, such as traceroute (37). Connections between the sender, routers, and
receivers are called *links*. Each link can be a direct connection between two routers, or there may be "hidden" routers (where no branching occurs). We perform link characteristic estimation on logical topology, by removing the "hidden" nodes along the path. Fig. 5.1 depicts an example of a logical tree-structured topology where a sender (node 0) is connected to 13 receivers (nodes 6 through 18). We adopt the notation that link $i$ connects node $i$ (below) to its parent node $h(i)$. Additionally, let $s(i)$ be the subpath from the root to node $i$, where $i$ is a node in the logical topology, i.e., an internal branching point or a receiver in the tree.

We assume that the routing table does not change during an experiment and that there is no load-balancing between the sender and the receivers during the measurement period. Thus the topology remains fixed and the paths are unique between the sender and each receiver. These are typical assumptions in network tomography studies. The assumption is in fact valid if the probing period is significantly short in comparison to the frequency at which the routing tables are updated (e.g., 15 minutes in (33)). This consideration imposes the constraint for short measurement period, typically less than 5 minutes, to ensure the routing table does not change.

The type of back-to-back probing described in Section 1.4, coupled with the tree-structured topology, leads to a linear system of measurement equations (74) (possibly after a linearizing transformation such as the logarithm) of the form

$$Y = AX + \epsilon$$
where $Y$ is the raw data acquired from end-to-end probing, $A$ is a $n \times n$ routing matrix determined by the topology, $X = [x_1, x_2, \cdots, x_n]^T$ is a $n \times 1$ vector of link characteristics (e.g., link delay variances, link loss rates) for links 1, 2, \cdots, $n$ and $\epsilon$ is an error term representing various sources of noise and randomness in the measurement process.

We model the errors as a zero-mean Gaussian random vector with covariance matrix $\Sigma$. Each element of $Y$ corresponds to the congestion experienced along a certain subpath. Therefore, we will refer to $Y_i$, the $i$-th entry of $Y$, as the data for subpath $s(i)$ (the path from node 0 to node $i$), $i = 1, \ldots, n$.

There are several assumptions in the framework that are worthy of discussion. The underlying measurement framework assumes that link characteristics are stationary during the measurement period. Furthermore, packets have similar experiences on each shared link. For example, the packet pair measurements to receiver 6 and 7 have similar experiences on common links, e.g., links 1 and 2 as shown in Fig. 5.1. Typical measurements on the Internet indicate that closely spaced packets are highly correlated. In this chapter, we assume packets within a pair are perfectly correlated on all shared links. We also assume spatial and temporal independence. This implies the link characteristics on neighboring links are not correlated as a result of cross traffic and successive packets across the same link would have the same link experience. These assumptions can be relaxed, both in theory and in practice (5), but such extensions are not central to the main points in this paper.

We assume that the measurements are statistically independent of each other, and therefore $\Sigma$ is diagonal with entries equal to the variance of the data $Y_1, \ldots, Y_n$, respectively.
Specifically, let $\sigma_i^2$ denote the variability of a single back-to-back probe measurement associated with $Y_i$, for $i = 1, \ldots, n$. If $k_i$ probe measurements are averaged to form the final statistics $Y_i$, then $\Sigma = \text{diag}(\sigma_1^2/k_1, \ldots, \sigma_n^2/k_n)$. The main question addressed in this paper is the following. \textit{Given a total budget of $N$ probes (i.e., $\sum_{i=1}^{n} k_i = N$), how should these probes be distributed among the different subpath measurements $Y_i$ to minimize the overall error of the network tomography experiment?}

### 5.2 Problem Formulation

We commence the description of the objective function to be minimized by observing the structure of the solution to a least squares estimation problem. Considering the general system of linear equations

$$Y = AX + \epsilon,$$

where the dimension agrees. The least squares solution to $Y = AX$ is

$$\hat{X} = \underbrace{(A^T A)^{-1} A^T Y}_{B}$$

Note that it is an unique minimizer of $||AX - Y||_2^2$. We simplify the notation by letting $B = (A^T A)^{-1} A^T$. Before we proceed further, let us first investigate the structure of the routing matrix $A$.

\textbf{Lemma 5.1.} \textit{Every routing matrix, $A$, constructed from a single sender, multiple receiver}
tree-structured network can be organized into a lower triangular form, with rank $n$.

Lemma 5.1 is obvious if we label all the links in such a way that the link ID of a child is always greater than that of its parent and all the nodes with a smaller depth from the sender. In other words, order the links in a lexicographic manner. For other constructions, we can apply the permutation matrix to reorder the rows, and the same result can be obtained.

Notice that for a non-load-balancing network, the routing matrix $A$ is always a binary matrix of size $n \times n$ for a $(n + 1)$ node tree-structured (acyclic) network. Furthermore, this matrix in lower triangular form will have ones as its main diagonal elements, thus the determinant is always one.

**Theorem 5.2. Theorem 2.5 quoted from (75)] If $A$ is a lower (upper) triangular with ones as its main diagonal elements then $A$ is non-singular and $A^{-1}$ is also lower (upper) triangular with ones as its main diagonal elements.

From Lemma 5.1 and Theorem 5.2, it is obvious that $B = A^{-1}$. The minimizer becomes simply

$$\hat{X} = BY = A^{-1}Y.$$  

The notation of $B$ and $A^{-1}$ will be used interchangeably in the rest of the paper. The variance of the estimates $\text{cov}(\hat{X}) = B\Sigma B^T$. Given that $\Sigma$ is diagonal, the variance of the estimates depends not only the variance on each link by also those along the path. Our goal is to reduce the variability by allocating the probes in an optimal manner.
5.3 Optimal Experimental Designs

In optimal experimental designs (76), one would typically summarize the quality of the design by a single number by either maximizing or minimizing on the design. The goal typically is to minimize the variance of the estimates or to maximize the information matrix given by (1/variance). Among different criteria, \textit{A-optimality} and \textit{D-optimality} are most commonly used. The A-optimality is to minimize the variances of the estimates, which is the trace of the \text{cov}(\hat{X}) in this case. The D-optimality is to maximize the determinant of the information matrix, which is the reciprocal of the determinant of the matrix \text{cov}(\hat{X}) in this case*.

\begin{align*}
\text{A-Optimality:} & \quad \min \quad \text{trace}(\text{cov}(\hat{X})) \\
\text{D-Optimality:} & \quad \max \quad 1/|\text{cov}(\hat{X})|
\end{align*}

We focus on A-optimality condition in this paper. It can be shown later that it provides explicit forms for the squared error of the link estimation errors in terms of topology, noise-levels and number and distribution of probes. The D-optimality condition, on the other hand, does not rely on the topology and is achieved as the result of current practice, as shown below. We will first look at the D-optimality condition, follow by more discussion on A-optimality, which is the focus of this paper.

*Assuming jointly Gaussian errors
5.3.1 D-Optimality

The D-optimality is to maximize the determinant of the information matrix, which is the reciprocal of the determinant of the matrix \( \text{cov}(\hat{X}) \).

\[
\frac{1}{\left| \text{cov}(\hat{X}) \right|} = \frac{1}{|B\Sigma B^T|} = \frac{1}{B|\Sigma||B^T|} = \frac{1}{|\Sigma|} = \prod_{i=1}^{n} \frac{k_i}{\sigma_i^2}
\]

The above equalities hold as a result of the properties of determinant. The determinant of a triangular matrix equals to the product of the diagonal entries. In our case, \( A \) and \( A^{-1} \) are both lower triangular matrix (Theorem 5.2) with diagonal entries equal to one, thus their determinants are both one.

The D-optimality condition is

maximize \( \prod_{i=1}^{n} \frac{k_i}{\sigma_i^2} \)

subject to \( \sum_{i=1}^{n} k_i = N, k_i \geq 1 \)

Since logarithm is a monotonic function, we could rewrite the objective function as
follow:

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \log k_i - \log \sigma_i^2 \\
\text{subject to} & \quad \sum_{i=1}^{n} k_i = N, k_i \geq 1
\end{align*}
\]

which is equivalent to

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \log k_i \\
\text{subject to} & \quad \sum_{i=1}^{n} k_i = N, k_i \geq 1
\end{align*}
\]  \quad (5.1)

Simple computation shows that the function is optimized when the resources \( N \) probes are evenly distributed among \( n \) nodes. Therefore, with D-optimality condition, the probe allocation problem simply implies distributing the resources evenly to all nodes. It depends only on the variability in the measurements but is invariant to connectivity. If \( N \propto n \), then it is optimal to have \( k_i = N/n \). The condition for \( k_i > 1 \) is automatically imposed.

### 5.3.2 A-Optimality

The A-Optimality condition relates the topology, noise levels and probes in the squared error computation of the network tomography problem. Before we proceed further, let us first introduce the term *outdegree*.

**Definition 5.3.** The outdegree of node \( i \) denoted \( d(i) \), is defined to be the number of directly connected links from node \( i \). It is also the number of children links connected to link \( i \) plus one.
In a logical tree-structured topology, the sender and receiver nodes have outdegree equal to one while the rest of the nodes have outdegree greater than 2. For example, in Fig. 5.1 the outdegree of node \( d(2) \) is 5.

In our case, the A-optimality is given by

\[
\text{trace}(\text{cov}(\hat{X})) = \text{trace}(B\Sigma B^T) = \text{trace}(\Sigma B^T B) \quad (5.2)
\]

The second equality holds since a cyclic permutation of the terms in the product preserves the trace, e.g. \( \text{trace}(ABC) = \text{trace}(CAB) = \text{trace}(BCA) \) when the dimensions agree. Recall that the minimizer is

\[
\hat{X} = BY = A^{-1}Y.
\]

The inverse of the matrix \( A \) can be computed using Gauss-Jordan elimination, Gaussian elimination or the LU factorization (77).

**Lemma 5.4.** The squared error for network tomography estimation or A-optimality criterion is \( \sum_{i=1}^{n} \frac{d(i)\sigma_i^2}{k_i} \), where \( d(i) \) is the outdegree of node \( i \), \( \frac{\sigma_i^2}{k_i} \) is the variance of the statistics \( Y_i \).

**Proof of Lemma 5.4** The above lemma is the result of direct computation of \( \text{cov}(\hat{X}) \). The proof is trivial and is included for the sake of completeness. If we have element-wise comparison between the row of node \( i \) and that of its parent node \( h(i) \) in matrix \( A \), the
difference is at matrix entry \( a_{i,j} \). Denote row \( i \) of matrix \( A \) as \( A(i,:) \), we have

\[
A(i,:) = \begin{cases} 
  +1 & : s(i) = 1 \\
  0  & : \text{otherwise}
\end{cases}
\]

The index for those links along subpath \( s(i) \) has ones, and zero otherwise. By computing the inverse of matrix \( A \), e.g. by using the Gaussian elimination, we have

\[
A^{-1}(i,:) = \begin{cases} 
  +1 & : i = 1 \\
  -1 & : h(i) = -1 \\
  0  & : \text{otherwise}
\end{cases}
\]

This implies

\[
b_{i,j} = \begin{cases} 
  +1 & : i = j \\
  -1 & : j = h(i) \\
  0  & : \text{otherwise}
\end{cases}
\]

The weight \( B^T B \) of the each diagonal entries \( \Sigma \) is the number of non-zero elements in each column of \( B \). In each column of \( B \), there is a +1 at node \( i \) and -1 at its child node(s). Therefore the weight is simply given by the outdegree, the number of child nodes plus number of parent node (i.e. 1). This completes our proof.

\[ \blacksquare \]
The objective function given the A-optimality is thus given by

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{n} \frac{d(i)\sigma_i^2}{k_i} \\
\text{subject to} & \quad \sum_{i=1}^{n} k_i = N
\end{align*}
\] (5.3)

where \( N \) is the probe budget that we wish to optimally allocate among the \( n \) measurements, \( Y_i, i = 1, \ldots, n \).

The optimality condition is not immediately obvious from Equation 5.3. However, it can be read as to distribute the total budget \( N \) to each links depending on the noise level as well as the outdegree (number of children links).

In the following section, we consider the objective optimization function, i.e. solving the optimization problem in Equation 5.3 and 5.1. This objective can be summarized as minimizing the squared error of the network tomography estimation.

### 5.3.3 Constrained Optimization

Recall that the objective function for probe allocation problem for both A-optimality and D-optimality condition is to minimize the squared error of the network tomography estimation. Both of them can be generalized as follow:

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{n} f_i(k_i) \\
\text{subject to} & \quad \sum_{i=1}^{n} k_i = N
\end{align*}
\] (5.4)
where \( f_i(k_i) = \frac{\sigma_i^2 d(i)}{k_i} \) or \( f_i(k_i) = -\log(k_i) \) given A- or D- optimality condition respectively, and \( N \) is the probe budget that we wish to optimally allocate among the \( n \) receivers, \( Y_i, i = 1, \cdots, n \). We have noted earlier that the condition that \( k_i > 0 \) is automatically imposed.

The optimization for D-optimality condition is trivial, thus we will focus in A-optimality. For the A-optimality condition, it depends on the quantity given for \( \sigma_i^2 \). Unfortunately, in most of the cases, \( \sigma_i^2 \) is not known \textit{a priori}. We will assume \( \sigma_i^2 = \text{constant}, \forall i \) for now and will relax the assumption to allow for heterogeneous noise power \( \sigma_i^2 \) later. This assumption allows us to illustrate the main concept and to demonstrate the optimality of the algorithm which will be described next.

Here, we note that our objective function is a discrete resource allocation problem. Moreover, the objective function is a convex function \( f_i(k_i) = d(i)/k_i \) in \( k_i \). The objective function can be optimized using the “incremental” algorithm that will be described next. This algorithm is simple and intuitive. We independently “discovered” the algorithm and later become aware of the fact that the same algorithm has been rediscovered by others from time to time (78). The algorithm was first described and discussed by Fox in 1966 (79) based on a paper of Everett in 1962 (80). Since then, there are numerous modifications and variations to further reduce the complexity of the algorithm. Refer to (78) for proofs and further details.

Here we present the “incremental” or the “marginal allocation” algorithm and briefly discuss the optimality of the approach.
1. Initialize \( k_i(t) := 0, \forall i \) at iteration \( t = 0 \).

2. Increment \( t \) by 1.

3. Find \( i^* \) such that \( f_i(k_i^* + 1) = \min_{1 \leq i \leq n} f_i(k_i + 1) \) and assign \( k_i^* = k_i^* + 1 \).

4. If \( t = N \), then stop (the current \( k \) is an optimal solution), else go to Step 2.

At the beginning of the algorithm, we first initialize \( k_i = 0, \forall i \). At each iteration, we assign one unit of the resource to the most favorable node \( k_i \), i.e., the smallest \( f_i(k_i) \) in which by increasing \( k_i \) of the objective function is minimized. We repeat the algorithm for \( N \) iterations, or until all of the resources are allocated. For any given \( N \), the algorithm provides an optimal and feasible solution.

Notice that our objective function is a discrete optimization problem with one constraint. This problem falls into the framework of Generalized Lagrange Multiplier (GLM) method or the “Marginal Analysis” (79; 80). The GLM is a useful technique and it can be applied to any strategy sets, objective functions and constraints, as long as they are real-valued. In the case where the objective function is convex and separable, the solution derived from GLM is shown to be optimal (80). The algorithm described above is a direct implication of the GLM which we will describe next.

5.3.4 Generalized Lagrange Multiplier

In most textbooks, the Lagrange Multiplier is introduced as an approach to solve extremum problems with constraints. These optimization problems are typically continuous and dif-
ferentiable. In Everett (80), it was further pointed out that the Lagrange Multiplier Method can also be considered as maximizing a function with constraints where there are no restrictions imposed on the functions. In cases where the function is concave and separable, the solution found is guaranteed to be optimal. When the concavity or separable conditions are violated, the approach is still useful by providing a bound on the error of the results.

We first quote the main theorem given in Everett (80) and recast our problem to show that the solution is indeed optimal. Define an arbitrary set \( S \) as the set of strategies. On this set, define a real valued payoff (objective) function \( H \), where \( H(x) \) is the payoff results from applying strategy \( x \in S \). We also define \( n \) real valued resource functions \( C_j^j(j = 1, \cdots n) \) on \( S \). For \( j \)th resources, we utilize \( C_j^j(x) \) resources. In this framework, the objective function is to maximize the payoff subject to given constraints \( c^j, j = 1, \cdots , n \) on each resource.

\[
\max_{x \in S} \quad H(x) \quad \quad (5.5)
\]

subject to \( C_j^j(x) \leq c^j, \quad \forall j \)

The main theorem is given as follow:

**Theorem 5.5.** [Lagrange Multiplier] Define \( \lambda^j, j = 1, \cdots , n \) are nonnegative real numbers. If \( x^* \in S \) maximizes the function

\[
H(x) - \sum_{j=1}^{n} \lambda^j C_j^j(x) \quad \forall x \in S,
\]
then \( x^* \) maximizes \( H(x) \) over all those \( x \in S \) such that \( C^j(x) \leq C^j(x^*) \) for all \( j \).

Theorem 5.5 illustrates the basic idea of GLM. It states that if a strategy that maximized the unconstrained objective function can be found, then this strategy is the solution to the original constrained objective function. This strategy produces the unconstrained maximum, i.e., the greatest payoff without using more resources than required by this strategy.

Now consider a subclass of the above problem, a separable/cell problem, where the resources will be distributed independently to each cell and the total payoff will be the sum of the payoffs from each cell. For each cell \( i \), define a strategy \( S_i \), a payoff function \( H_i \) defined on \( S_i \) and \( m \) resource functions \( C_i^j \) defined on \( S_i \). This problem is a subclass of the previous framework, where

\[
S = \prod_{i=1}^{m} S_i.
\]

The modified objective function becomes

\[
\max_{\text{all choices of } x_i, x_i \in S_i} \sum_{i=1}^{m} H_i(x_i)
\]

subject to

\[
\sum_{i=1}^{m} C_i^j(x_i) \leq \xi^j, \forall j
\]

In other words, the unconstrained Lagrange function is

\[
\max_{x \in \prod_{i=1}^{m} S_i} \left[ \sum_{i=1}^{m} H_i(x_i) \right] - \sum_{j=1}^{n} \lambda^j \left[ \sum_{i=1}^{m} C_i^j(x_i) \right],
\]
which is equivalent to

$$\max_{x \in \prod_{i=1}^{m} S_i} \sum_{i=1}^{m} \left[ H_i(x_i) - \sum_{j=1}^{n} \lambda_i^j C_i^j(x_i) \right]$$

by interchanging the summation order.

The cells are independent from each other and the problem is thus uncoupled. The result from Theorem 5.5 guarantees that if each cell is correctly maximized, the result to the overall problem is also globally maximized.

Recall our objective function is to allocate the given number of probes, $N$, to $n$ nodes in order to minimize the total cost. It is equivalent to maximize the negative of the total cost. The discussion above holds without loss of generality. Each probe allocation can be considered as a cell in our previous discussion. Our objective function can now be rewritten as

$$\min_{i=1}^{N} \sum_{i=1}^{N} \left[ -H_i(x_i) + \sum_{j=1}^{n} \lambda_i^j C_i^j(x_i) \right]$$

For each probe $i$, we want to minimize the current cost function $H_i(x_i)$ by choosing the smallest $f_j(k_j), j = 1, \cdots, n$ nodes. The resource function $C_i^j$ for node $j$ and $i$th probe is thus $C_i^j = \min_{1 \leq j \leq n,i} f_j(k_j)$. When there are more than one $j$ satisfies the minimum, we further impose $\lambda_i^j$ to be integer, thus only one $j$ will be chosen at each $i$. In other words, all $\lambda_i^j$ will be zeros except one of them equals to 1. The “incremental” algorithm is a direct implication of the Generalized Lagrange Multiplier method. Furthermore the cost function is convex in $k$, thus the solution given by GLM is indeed optimal as our problem at hand is
5.4 Scalability Study

Results from optimal probe allocation problem also provide insight into the scalability of network tomography estimation. For simplicity and without loss of generality, we assume the total budget (measurement probes) $N$ to be multiple of the number of nodes $n$, i.e., $N = mn$ where $m \geq 1$ is a constant.

In order to bound the squared error in Lemma 5.7, we first analyze optimal probe allocation when the outdegrees of each link are equal.

**Lemma 5.6.** If the outdegree for all nodes are the same, $d(i) = c \forall i$, where $c = \text{constant}$, then uniformly distributing the probes among receivers is optimal.

**Proof of Lemma 5.6** Given $N$ measurements and $n$ nodes, we want to show that $k_i = N/n$ for $i = 1, \cdots, n$. The outdegree of each node is $d(i) = c, \forall i$. The squared error which is the objective function for optimization becomes,

$$
\text{minimize } \sum_{i=1}^{n} \frac{c}{k_i}
$$

subject to $\sum_{i=1}^{n} k_i = N$

The optimization problem can be solved using a Lagrange multiplier, $\lambda$, to minimize

$$
L(\lambda) = \sum_{i=1}^{n} \frac{c}{k_i} + \lambda \left( \sum_{i=1}^{n} k_i - N \right)
$$
by finding the $\lambda^*$ such that $\frac{\partial L(\lambda^*)}{\partial k_i} = 0$. Note that $\lambda^*$ is the unique minimizer as each $f_i(k_i)$ is strictly convex. The optimal solution is $k_i = N/n$, $\forall i$.

We will apply the result of Lemma 5.6 in the computation of the bounds of the squared error.

**Lemma 5.7.** Define $d_{\text{max}} = \max\{d(i)\}$ and $d_{\text{min}} = \min\{d(i)\}$ for $i = 1, 2, \ldots, n$. The minimum squared error per receiver of the network tomography estimation can be bounded as follows:

$$\frac{d_{\text{min}}}{N^r} \leq \min_{k_i; \sum k_i = N} \left\{ \frac{1}{r} \sum_{i=1}^{n} \frac{d(i)\sigma_i^2}{k_i} \right\} \leq 4 \frac{d_{\text{max}}}{N^r}$$

The number of probes required to achieve a given level of accuracy per receiver grows linearly with the number of receivers involved in the experiment.

**Proof of Lemma 5.7** Considering the squared error for all receivers. First, let’s look at the upper bound of $\min_{k_i; \sum k_i = N} \left\{ \frac{1}{r} \sum_{i=1}^{n} \frac{d(i)}{k_i} \right\}$. Applying Lemma 5.6 and noting that
\( r \leq n \leq 2r \) (where \( r \) is the number of receivers), we have

\[
\min_{k_i: \sum k_i = N} \left\{ \sum_{i=1}^{n} \frac{d(i) \sigma_i^2}{k_i} \right\} \leq \sum_{i=1}^{n} \frac{d_{\max}}{k_i} = \sum_{i=1}^{n} \frac{d_{\max}}{N/n} = \frac{d_{\max}}{N} n^2 \leq \frac{d_{\max}}{N} (2r)^2
\]

Similarly for the lower bound

\[
\min_{k_i: \sum k_i = N} \left\{ \sum_{i=1}^{n} \frac{d(i)}{k_i} \right\} \geq \min_{k_i: \sum k_i = N} \left\{ \sum_{i=1}^{n} \frac{d_{\min}}{k_i} \right\} \geq \sum_{i=1}^{n} \frac{d_{\min}}{N/n} = \frac{d_{\min}}{N} n^2 \geq \frac{d_{\min}}{N} r^2
\]

Define \( e^2 = \min_{k_i: \sum k_i = N} \left\{ \frac{1}{r} \sum_{i=1}^{n} \frac{d(i) \sigma_i^2}{k_i} \right\} \). The squared error per receiver is thus

\[
c_1 r \leq e^2 \leq c_2 r
\]

where \( c_1 \) and \( c_2 \) are constants and \( c_1 < c_2 \). This completes our proof.
5.4.1 Heuristics for Computing Unknown Error

In the previous section, we assume the noise power $\sigma_i^2 = \text{constant}$. We now relax the objective function to allow for heterogeneous noise power. The algorithm remains unchanged and the solution to our optimization remains optimal and feasible for each given budget of $N$ measurements.

In this section, we introduce several heuristics for estimating the unknown error term, $\sigma_i^2$.

**Hop counts:** The noise variance on each link depends on the amount of cross traffic, i.e. the variability increases as the number of hops (number of actual routers along the path, instead of the branching router in the logical topology) increases. At each router, the intervening traffic induces extra delay and chances of being dropped.

**Pre-measurement:** Before carrying out the actual measurements, send small number of probes to estimate the variability.

**Two-stage:** Apply a two stage procedure, use half of the available probes to estimate the variability, then apply the remaining half according to the algorithm described above.

5.5 Simulation Experiments

We collected end-to-end delay statistics from all packet pair measurements using the *ns-2* simulation environment. Then, we apply our optimal probe allocations using different estimates of $\sigma_i^2$. Finally, we explore the squared error with different probe allocation strategy
with various measurement budgets $N$. 

5.5.1 Simulation Framework

We use the topology as shown in Fig. 5.1 for our simulation studies. The topology is intended to reflect (to some extent) the nature of many networks, where the edge links are slower and the internal links are the backbone links which are faster. The topology also includes a hidden node in which cross traffic can also enter the network through a non-branching node. The outdegree ranges from 3 to 5 for the branching nodes. We fix the size of all queues to be 35 packets and using the Droptail policy. In all experiments, we assume there are long TCP (FTP sessions) connections to the receivers that last for the extent of the measurement period. In addition, there are a variety of short duration TCP sessions randomly set up and torn down between some random combinations of nodes. The average utilization of the network is relatively high in order to have non-zero queueing delays. We set up the packet pair spacing to be 33ms.

5.5.2 Simulation Results

We conduct ten independent simulations over a measurement period of 120 seconds. We examine the squared error from uniform or optimized probe allocations varying the budgets of $N$ probes.

- **Uniform** probing using $k_i = N/n$ and $\sigma_i^2 = \text{constant}$
- **Optimal** probing where \( k_i \) derived from incremental algorithm and setting \( \sigma_i^2 = \hat{\sigma}_i^2 \)

where \( \hat{\sigma}_i^2 \) is an estimate computed from the \( N/2 \) measurements

The squared error arising from uniform and optimized probing are compared in Fig. 5.2.

![Graph comparing squared error with uniform and optimal probing schemes](image)

**Figure 5.2**: Comparison of the squared error given a probe budget with uniform and optimal probing schemes.

The error decreases as the number of probes increases, as expected from Fig. 5.2 network tomography estimation. Besides, the optimal probing scheme outperforms the uniform scheme.

### 5.6 Related Work

There is a vast literature on network tomography (1; 2; 3; 5; 6; 11; 13). Most of the previous work focuses on congestion detection by estimating the loss rates or delay distribution functions (1; 4; 5; 11; 12). There is also some work on inferring the topology under study (4; 12; 81). Inference of link characteristics is based on probes that are actively
injected into the network from one sender to multiple receivers using unicasting or multicast probing (20). Passively monitoring packets for inference is also considered in (2; 6).

Whether to probe with unicasting or multicasting depends on the application under study. However, most traffic in the Internet is unicasting in nature and the routers respond to multicast packets in a different manner (20). Therefore, most recent work focuses on unicasting probing. The drawback of injecting unicasting probes as compared to multicasting probing is the increase in network load. In unicasting, the network load grows linearly with the number of nodes $n$, while in unicasting the load increases as $n^2$.

Back-to-back (closely time spaced) packet pair probing techniques were first used for congestion control in (82). Using packet pair in unicasting probing imitates the correlation of packets in multicasting probing. Suppose two closely time spaced packets are sent from the source to two different receivers. The paths to these receivers initially traverse a common set of links, but at some point the two paths diverge (as the tree branches). The two packets should have similar experiences on each shared link in their path. This observation has been experimentally verified in real networks (33). Other probing techniques have also been proposed, such as the packet stripes (10) in link loss inference and the sandwich probe (4) in topology identification.

There is also some research into improving estimation by studying temporal dependence in packet loss (83), and by estimating and removing clock skew in delay estimation (39; 40). Network tomography techniques are also being recognized and applied in sensor networks (84).
Network resource allocation has been studied in various forms of the waterfilling problem for sharing resources among multiple users, such as processor sharing in distributed computing, power control in wireless networks and distributing resources to different classes in Quality of Service studies. See (85) for general references.

5.7 Summary

We have demonstrated that optimized allocation of network probes can significantly improve network tomography. In the process of developing our strategies for optimized probing, our error analysis revealed the dominant sources of estimation error: network topology and measurement error. Topology is generally assumed to be known in network tomography, so this information can be used to help guide probe allocation. Furthermore, estimates of measurement errors can be gleaned from an initial set of probes and used to optimized the allocation of subsequent probes. We have demonstrated the capability and improvement in the estimation using a simple discrete resource allocation problem algorithm, namely the "incremental" algorithm. The algorithm is simple and yet, provides an optimal solution with $N$ measurements to the probe allocation problem. Our analysis also indicates the number of probes required for a given level of accuracy. Our experiments demonstrate the potential of the new methods.
Chapter 6

Conclusion

This thesis has focused on fundamental theory and methods for network monitoring, the development of practical measurement schemes, and algorithms for performance estimation. In particular, two important performance parameters, link loss rates and link delay distributions, are studied using only end-to-end measurements.

The link-level performance information provides a local measure of the congestion at internal points in the network. We address some of the important issues commonly faced in these parameters estimation problems. In link loss rate inference, we propose a passive monitoring technique to estimate drop rates on internal network links that only requires TCP traces from the endhosts. This approach is more effective and less intrusive than most other tomography schemes. Recognizing data starvation and scalability as some of the potential limitations of passive measurement schemes, we propose alternative data-mining and servicing strategies at the source that may provide more informative data. We also discuss a method for clustering receivers to reduce the effective complexity of the network, thus allowing us to focus on identifying loss rates on a more relevant subset of links.

In the delay case, I developed a non-parametric approach for estimating link-level queuing delay characteristics. The approach overcomes the bias-variance tradeoff caused by delay quantization, a problem associated with most existing delay estimation methods. Some
of the key features of our framework are its ability to capture fine details and smooth regions, and the introduction of a complexity penalization that allows smooth, accurate estimates to be generated even when the amount of data is very small. The basic MPPLE framework developed here could be extended to the multicast approach suggested in (10) and may also be applicable in time-varying contexts like those considered in (8; 9).

We have also addressed some of the fundamental challenges faced in deploying our inference techniques as well as those developed by the others. Most proposed network tomography schemes are not directly deployable or broadly applicable because of practical limitations, such as the need for cooperation between end-hosts, the lack/difficulty in synchronizing clocks across systems, and the scalability of probing schemes and algorithms. This study allows the inference techniques to be more practical and reliable.

The novel technique based on Round Trip Time measurements, does not require clock synchronization and does not require special-purpose cooperation from receivers. It is similar to diagnostic tools like traceroute but unlike traceroute, it provides localized one-way information and can be used for link-level queuing delay characterization and network topology identification. The RTT method is shown to be effective under a wide range of operating conditions. While the implications of this work are that Network Radar is a technique that could be used widely, some additional work would make it even more applicable. First, we are continuing to evaluate and refine our inference methods with the objective of further reducing the number of probes used to establish delays on a given shared segment. Second, it is straightforward to extend our tool to additionally output packet loss
estimates thus enabling highly congested links to be isolated and identified. Finally, we plan to implement our techniques in a stand alone tool that can be distributed, and that runs on several common operating systems (currently the measurement and inference tools are separate).

In the scalability study, instead of studying the problem directly, the network tomography problem is transformed into an optimal experimental design problem. This perspective provides significant new insights on the efficiency and scalability of tomographic techniques. On-line and distributed versions of the optimized probing algorithm are potentially fruitful avenues for future work.
Bibliography


