Theory and Practice of Non-Intrusive Active Network Measurements

Sridhar Machiraju

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2006-68
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-68.html

May 18, 2006

Theory and Practice of Non-Intrusive Active Network Measurements

by

Sridhar Machiraju

M.S. (University of California at Berkeley) 2003

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Randy H. Katz, Chair
Professor Ion Stoica
Professor Charles Stone

Spring 2006

The dissertation of Sridhar Machiraju is approved.

Chair Date

Date

Date

University of California, Berkeley

Spring 2006

Abstract

Theory and Practice of Non-Intrusive Active Network Measurements

by

Sridhar Machiraju

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Randy H. Katz, Chair

Today's data networks are highly distributed and enormous in scale. The ability to measure them is vital to both network operators and end-users. Network measurement methods can broadly be classified into *passive methods* that rely on data collected at routers, and *active methods* based on observations of actively-injected probe packets. Active measurements, the focus of this dissertation, are attractive to end-users who, under the current network architecture, cannot access any measurement data collected at routers. Network operators use active measurements because they are easy to conduct, have low overhead and, in contrast to passive data collection methods, measure exactly what normal data packets experience. The most significant disadvantage of active measurements is the limited accuracy that has typically been achievable using them. One of the main reasons for this is in the need to be non-intrusive, thus leaving the measured systems uninfluenced by the observation, fundamentally affecting accuracy.

In this dissertation, we use rigorous theoretical analysis to understand the impact of non-intrusiveness on active measurements and investigate how this theory translates into practice. Our investigation consists of three parts. In the first, we investigate sampling-related issues, i.e., when do we send probe packets and why? Our starting point is conventional wisdom that says that the "Poisson Arrivals See Time Averages (PASTA)" principle implies the need to use Poisson probing. We show that PASTA does not imply that Poisson probing is optimal because it ignores bias caused by

probing intrusiveness and estimation variance. Using rigorous theory and simulations, we motivate rare probing, preferably at so-called *mixing* epochs, as a sound practical strategy. In the second part, we investigate if observed delays of (non-intrusive) probe pairs can be used to *estimate cross-traffic* properties in the single-hop case. Our starting point is the inability of prior works [SKK03] to estimate cross-traffic without hard-to-achieve timing control. We derive what can be estimated, in theory, and show that, under a well-motivated assumption, non-intrusive probe pairs can be used to estimate the entire distribution of cross-traffic in an intra-pair interval. Our third part is motivated by the apparent difficulty in designing non-intrusive active measurements robust to multi-hop queueing effects; We experience this first-hand with our single-hop cross-traffic estimators. We show that novel hop-dependent priority queueing primitives can be used to design *Measurement-Friendly Networks (MFNs)*, networks in which accurate non-intrusive measurements, which are robust to multi-hop queueing effects, can be performed. Our primitives not only simplify network management tasks for network operators but are also easily deployable. In exploring MFNs, we find that nonpreemption and cross-traffic persistence cause unavoidable inaccuracies that represent, in a sense, fundamental limitations of active measurements.

Professor Randy H. Katz
Dissertation Committee Chair

To my dearest Amma and Nanna who have always been there for me,

and to Snigdha, the love of my life.

Contents

List of Figures

List of Tables

Acknowledgements

This dissertation would not have been possible without my advisor Randy. He continued to be patient throughout and constantly encouraged me. In spite of his busy schedule, he made sure that we met every week. Towards the end, he constantly took pains to ensure that I did not neglect the laborious process of putting the dissertation together and gave remarkably detailed feedback within a few days. While working on this dissertation, I had the pleasure of collaborating with three truly wonderful researchers - Jean Bolot, Darryl Veitch and François Baccelli. I am grateful to Jean for his enduring faith in me. He invited me to join his team at Sprintlabs and ensured that I had enough flexibility and resources to complete this dissertation. During brainstorming sessions he would get to the crux of the matter very quickly and, in his polite way, separate the good ideas from the bad. Working with Darryl was like working with a fellow graduate student. We argued, philosophized, discussed minute details and wrote code, most of the time using Skype! From Darryl I learnt the importance of thoroughness, attention to detail, statistical concepts and Matlab programming (his proselytization worked within 6 months). François is one of the most knowledgeable persons I have ever met. Not only is he brilliant but he is also someone who can explain concepts such as ergodic theory as simply as arithmetic. That he is so accessible and good-natured only makes every interaction with him a truly wonderful experience. His dedication to work is remarkable. Within a day of our discussing a problem, I would receive a 10-page summary. From my initial years at Berkeley, I have always enjoyed interacting with Ion Stoica. I am thankful for all the valuable advice he has given me - on giving talks, good work ethics, writing papers, etc. I would also like to thank other faculty members, at Berkeley and elsewhere, with whom I have had the opportunity to interact with - David Wagner, Rene Cruz, Don Towsley, Doug Tygar, Richard Karp.

Over the course of my time at Berkeley, I have had the opportunity to interact with so many great students. Memories of those experiences are very dear to me. I thank them for their conversations, trips to cafes, brainstorming sessions, outdoor activities at the retreats, feedback on research, etc. These students, in no particular order, are Matthew Caesar, Weidong Cui, Jimmy Shih, Mukund Seshadri, Karthik Lakshminarayanan, Ananth Rao, Lakshminarayanan Subramanian, George Porter, Sharad Agarwal, Yin Li, Ana Sanz Merino, Jayanth Kumar Kannan, Bhaskaran Raman, Fang Yu, Mel Tsai, Tal Lavian Sonesh Surana, Shelley Zhuang, Dennis Geels, Ling Huang and many others.

I am also thankful to Keith Sklower and Damon Hinson for their infrastructural and administrative help. Thanks are due to current and former researchers in the IP group at Sprint - Antonio Nucci, Supratik Bhattacharyya, Hui Zhang, Tao Ye and Ashwin Sridharan. They provided a lot of support, encouragement and feedback. I also thank Ed Kress and Jamie Schneider for their invaluable help in conducting my experiments.

Last but not least, I am indebted to my family for their love, affection and constant support. The lessons my parents taught in my childhood and the values they instilled in me have, during my time at Berkeley, helped me thorough many difficult times. My monthly visits to my sister's place provided me much-needed breaks from the ups and downs of graduate life. The food and entertainment that my sister, Sirisha, and her husband, Karthik provided, lifted my spirits on countless occasions. My heartfelt thanks to them. My fiancée, Snigdha, bore the brunt of my frustrations as this dissertation took shape. Without her encouragement, companionship, cheerful attitude, and food, I would have taken much longer to finish this dissertation! There are several others who directly or indirectly made my stay at Berkeley worthwhile and fun. My heartfelt gratitude to all of you.

Chapter 1

Introduction

"One good measurement is worth a thousand expert opinions."

- Grace Hopper, Rear Admiral in the American Navy and pioneer computer scientist.

The ability to measure computer systems is vital for a variety of functions including debugging, managing, optimizing, characterizing, and forecasting. The Internet is no exception. The scale and distributed nature of the Internet makes measurement very challenging. For instance, most Internet traffic traverses multiple domains on the path from the source host to destination host. Typically, these domains are controlled by different administrative entities each of which does not share intra-domain measurements with other domains. Hence, even on a well-instrumented network path, no single entity can access *all* the available measurement information. Moreover, the scale of most domains makes intra-domain instrumentation quite costly too. Active network measurement techniques have been explored as solutions to these unique challenges of the Internet. Active measurement techniques estimate network properties by analyzing the end-to-end delays and losses of *probe* packets actively injected into the network. Often, probe packets can be replaced by suitably chosen packets of ongoing data flows.

Since active measurement techniques inject additional (probe) traffic, they inevitably perturb the system being measured. Therefore, to minimize the impact of measurements on normal data traffic and to access properties of the unperturbed system, it is highly desirable to keep the measurements as *non-intrusive* as possible. However, there is typically a trade-off between non-intrusiveness and

measurement accuracy. For instance, doubling the probing rate to estimate, say, mean end-to-end delay on a path, improves estimation accuracy by doubling the number of samples at the cost of increased intrusiveness (this is not true for arbitrarily high probing rates, though [Rou05]). Recently [LRL05a] similar trade-offs between non-intrusiveness and accuracy have also been observed in active measurement techniques to estimate available bandwidth. *In this dissertation, we use rigorous theoretical analysis to understand the impact of non-intrusiveness on active measurements and how this theory translates into practice under real-world constraints.* Thus, we not only design practical techniques for use by end-users and network operators but also provide insights into when and why these work, which is of interest to researchers and network modelers developing newer techniques.

Our investigation is structured around a *sampling-inversion* viewpoint [Vei05] of active measurements. This viewpoint decouples the sampling-related problems - when to send probe packets - from those related to inversion - how to derive the desired network property from the observations collected from probe packets. Using rigorous statistical and probability theory, we investigate fundamental "how-to" questions that arise in the context of both sampling and inversion with non-intrusive measurements. In the process, we find that the multi-hop nature of network paths causes difficult challenges that are only exacerbated by the need to keep measurements as non-intrusive as possible. This motivates us to explore network primitives that address the twin challenges of multi-hop paths and non-intrusiveness thereby making it possible to implement a *Measurement-Friendly Network (MFN).* Our primary focus is on unicast measurements that measure the properties of a particular network path using observed *delays* of probe packets. Probe delays, along with losses, are the basic observations that can be obtained via active measurements. Our primary focus is on delays because they are simple, more basic (loss can be considered to be infinite delay) and have proved to be much more useful in accessing bandwidth-related path properties.

1.1 An Example

We illustrate the various issues that typically arise using an example active measurement problem. Consider a typical unicast path similar to the one shown in Figure 1.1. The path has multiple hops, each of which has cross-traffic traversing it. Notice that cross-traffic may *persist* for more than

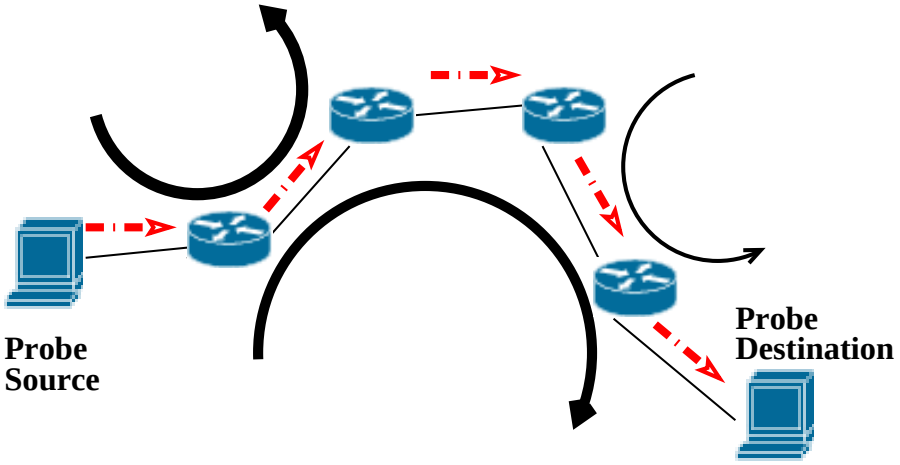

Figure 1.1. A canonical active network measurement setting. Probe packet streams (shown in red) are sent from the source to destination along the unicast path. They share the path with cross-traffic (dark solid arrows). The amount of cross-traffic at each hop may be different and is represented by the different thickness of the cross-traffic arrows. Cross-traffic may *persist* for more than one hop of the path.

1 hop along the path. The cross-traffic arrival processes are, therefore, the only time-varying inputs to the system. Knowledge of cross-traffic properties is, therefore, desirable to both end-users and network operators. End-users can use such knowledge for estimating a variety of metrics including available bandwidth [SKK03, JD03] and per-hop queueing delays [Dow99]. It can also be used to make decisions on the suitability of the path for jitter-sensitive applications such as Voice-over-IP. Knowledge of cross-traffic properties enables network operators to understand the queueing-related congestion at various hops, their root cause, alleviation strategies and so on.

Any technique to estimate cross-traffic properties has to use observations of suitable probe packet streams. Prior active measurements [Pax99, Kes95, SKK03, HS03, JD03] have used observations of delays experienced by probe packets, pairs and trains. Assume that we decide to use the observed delays of probe packet pairs for estimating cross-traffic properties. Having decided on estimating cross-traffic using probe pairs, we are faced with many questions. The most basic question is the in-principle potential, i.e., what properties of cross-traffic can we expect to access using probe pair delays? Some prior works [SKK03] estimated the average amount of cross-traffic in an intra-pair interval. Can we infer more, such as the entire Cumulative Distribution Function (CDF)? Additional questions relate to how we perform in-principle *inversion*, i.e., specific algorithms to access such cross-traffic properties using probe pair delays. Invariably, such inversion relies on a

well-motivated model of the path. One common assumption [SKK03, JD04] is to model the path as a single hop given that many Internet paths consist of a single predominant bottleneck. Since the ultimate goal is to estimate cross-traffic in real systems, we need to design practical techniques based on the in-principle inversion. In this context, the questions to be addressed include those related to parameter settings, model validation and performance comparison.

All of these questions relate to inversion, i.e., deriving cross-traffic properties from observed probe pair delays. So far, we ignored how we obtain the latter. For instance, are we justified in sending a periodic probe stream and using the delays of consecutive probes? A previous method [SKK03] applied the "Poisson Arrivals See Time Averages" (PASTA) [Wol82] principle to send probe pairs at Poisson epochs. They reasoned that Poisson pairs provide unbiased estimates. Is this use of PASTA valid and/or necessary? Bias is with respect to some underlying ground truth. Since probing perturbs the system, the ground truth is that of the perturbed system [Wol82]. What are the implications of this especially given the need for non-intrusiveness? All the above questions center around *sampling*, i.e., the times at which probe packets should be sent.

1.2 The Canonical Active Network Measurement Setting

In the previous section, we used an example to illustrate the various issues that arise in designing solutions to active measurement problems. We described them according to the sampling-inversion viewpoint [Vei05]. In the next section, we discuss broader versions of these issues that form the focus of this dissertation. First, in this section, we discuss preliminaries that help drive our discussion. These preliminaries clarify our scope and present relevant terminology and other important concepts such as non-intrusiveness and performance metrics.

Figure 1.1 is also the canonical setting for the measurement techniques that fall within our scope, namely, a unicast path that often consists of multiple hops. The goal of any active measurement technique is to estimate one of many possible path properties. We refer to this as the *desired* property. A typical desired property is the end-to-end delay that packets would experience along the path. This is of interest to both end-users and network operators. Another property of interest to end-users is the spare capacity also known as the available bandwidth. This is useful in

deciding the rate at which applications may inject packets; TCP slow start [Jac88] is a rather crude but widely-used method for such estimation. More sophisticated methods [JD03] are also known. Per-hop characteristics such as across-hop delays [Dow99], and busy periods [HVPD04] are also of interest especially to network operators.

As is clear from the above discussion, active measurement techniques are of interest to both end-users (e.g., individual hosts, overlays) and network operators for a variety of purposes. End-users typically use active measurements to optimize end-to-end performance [Jac88, ABKM01] since they have no access to performance data collected at routers. Researchers have also used active measurements to characterize network paths [Pax99, ZDP01, HLM+04]. Though network operators can deploy costly, high-speed passive data collection methods to debug, manage and characterize their intra-domain paths, they often use active measurements as lightweight alternatives that not only have very low deployment overhead but also directly measure what data packets experience [SBDR05]. Network operators also use active measurements to debug and characterize paths in other domains.This is done to obtain competitive information and also to verify service level agreements (SLAs) [JD03].

Active measurement techniques estimate the desired property by directly observing the delays or losses that individual probe packets, pairs or trains experience. In this dissertation, our primary focus is on using delays as the *observed* property. We do this because delays are simple and have proved to be more suitable for a variety of problems including bandwidth estimation. Delay is also a more fundamental quantity because loss can be considered to be infinite delay. In our example problem earlier, the observed property was the delays of probe pairs. Additional examples of observed properties are packet losses used by TCP slow start [Jac88] and end-to-end delays of probe trains [JD03]. The desired and observed properties may be the same when, for example, probe delays are used to estimate statistics of end-to-end delays [Pax99].

1.2.1 Solution Requirements

Performance of active measurement techniques is typically viewed in terms of the related metrics of accuracy, speed and overhead of estimation. Accuracy refers to minimizing estimation errors

that may be due to fundamental limitations of the technique such as modeling errors, or practical limitations like timing inaccuracies. Accuracy typically increases as more observations are made. Estimation overhead, a metric that refers to the number of probe packets required for some degree of accuracy, captures this. Speed combines overhead with the technique's sending rate and hence, is a metric that tells us how quickly estimation reaches some degree of accuracy. For instance, in our example problem, the difference between the estimated and true mean cross-traffic rate would be a suitable metric of accuracy. The number of probe pairs required to reach within, say, 5% of the true mean is the measurement overhead. The overhead divided by the specified rate of sending probe pairs is the speed of estimation. Notice that all three metrics are related.

In an ideal world, a measurement technique would be *non-intrusive*, i.e., the very act of measurement would not affect the system being measured. In reality, no active measurement technique is non-intrusive since all techniques inject probe packets of non-zero size. Nevertheless, to minimize the impact of measurement on existing data traffic and to fulfill the goal of measuring the unperturbed system, it is necessary to keep measurements as non-intrusive as possible. However, there is a fundamental trade-off between estimation performance and non-intrusiveness. This is easily seen by our earlier observation that estimation performance usually improves with more observations; With a fixed measurement duration, more observations can be obtained only by increasing the frequency of probing and hence, intrusiveness.

Intrusiveness has also been known to improve estimation robustness - a prime example being available bandwidth estimation. Techniques to estimate available bandwidth can be classified into those that measure it by saturating the bottleneck and observing the resulting queue buildups [JD03] and those that attempt to estimate it without causing any such buildups [SKK03]. We consider the former "fundamentally intrusive" because, unlike the latter, they intrinsically rely on causing larger delays for normal data traffic. Indeed, these fundamentally intrusive techniques are known to be more robust to unexpected queueing effects than the latter because the latter rely on fine-grained timing control. Even among the fundamentally intrusive techniques, recent work [LRL05a] observed that robustness improves with increased intrusiveness. We believe that, even though the fundamentally intrusive methods can be made effectively non-intrusive by spacing out the probe trains, it is preferable to not rely on queue buildups. Moreover, on distributed infrastructures such

as Planetlab [Pla] that have strict rate limits, it is not possible to use the large trains of probe packets required by the fundamentally intrusive techniques. Hence, in this dissertation, our primary focus is on measurements that are fundamentally non-intrusive and exploring the trade-off between performance and non-intrusiveness of such measurements.

1.3 Non-intrusive Active Network Measurements: Fundamental Questions

In the above, we motivated the various issues that arise in designing active measurement techniques using an example problem. In this section, we describe the broader set of questions implied by this example. These are the key questions addressed in this dissertation. The questions in this section are divided into three categories (see Figure 1.2). The first two categories are sampling (how to obtain the observed property) and inversion (how to obtain the desired property from the observed property). Recall that we encountered specific questions related to sampling and inversion while discussing our example problem. The third category of questions attempt to use network architecture design to understand how to tackle certain difficult challenges. These challenges, motivated by prior works and our investigations into sampling and inversion, relate to the multi-hop nature of paths and non-intrusiveness.

1.3.1 Sampling

While discussing our example, we alluded to prior work [SKK03] that used the "Poisson Arrivals See Time Averages (PASTA)" principle to send probe pairs at Poisson epochs. This work argued that, by PASTA, only Poisson probing leads to unbiased estimates. In a bid to encounter biased estimation, we conducted a simple experiment consisting of two simultaneous probe pair streams, one of which sent the pairs periodically and the other sent them at Poisson epochs. The intra-pair separation in both was 10ms and the experiment lasted for 3000s. We plot the "jitter", i.e., the difference between the delays of probe pairs, observed by both streams in Figure 1.3. The top plot shows a (vertically staggered) long-term average and a windowed average of the jitter. The

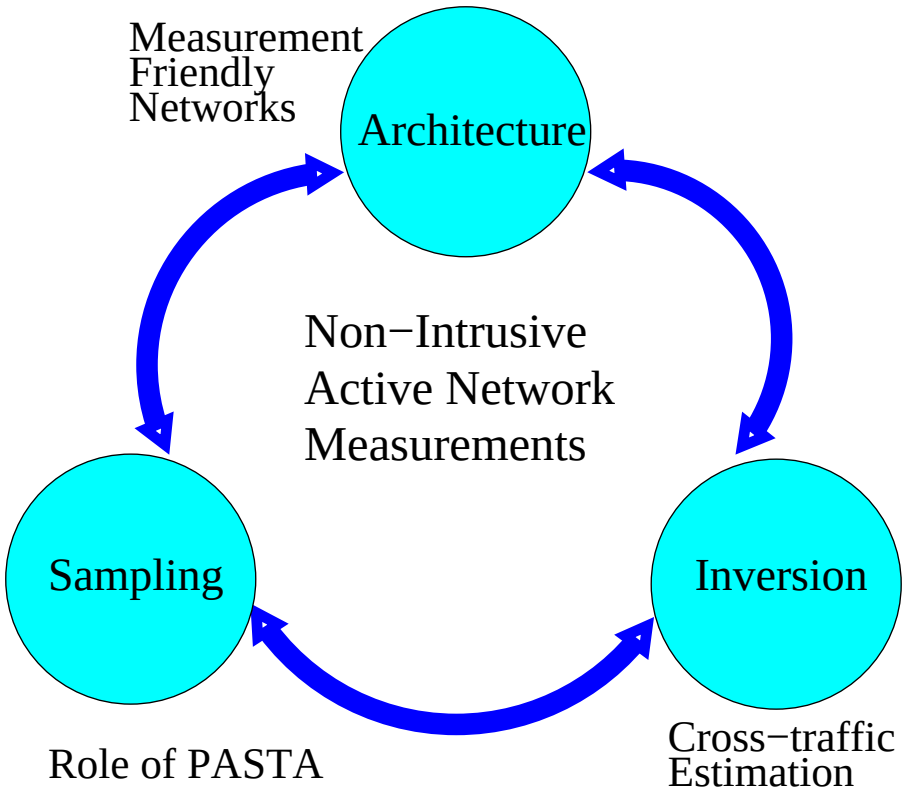

Figure 1.2. A schematic of the three categories of questions in non-intrusive active measurements that we address in this dissertation. Each category impacts the other. For example, the kind of inversion has an impact on the sampling strategy. As another example, fundamental limits on sampling may depend on the network architecture. We also summarize our specific contributions in each of these categories.

bottom plot shows the individual samples of jitter we obtained. We make two observations - the short-term averages were very close to each other except the portions where a large variance was observed. Moreover, the long-term averages coincided until the first region with large variance and did not match thereafter; However, the difference is only within a few hundred microseconds. Figure 1.3 raises many questions regarding the optimal sampling strategy especially the role of PASTA. A previous more-detailed empirical study [TDDA05] of periodic and Poisson probing also raised questions regarding the relevance of PASTA.

The above discussion motivates the following questions, answers to which allow us to decide how we obtain (sample) the observed property in active measurements. We start with the more basic questions and move to the more detailed questions.

- What is the measure of sampling performance, i.e., how do we compare different sampling

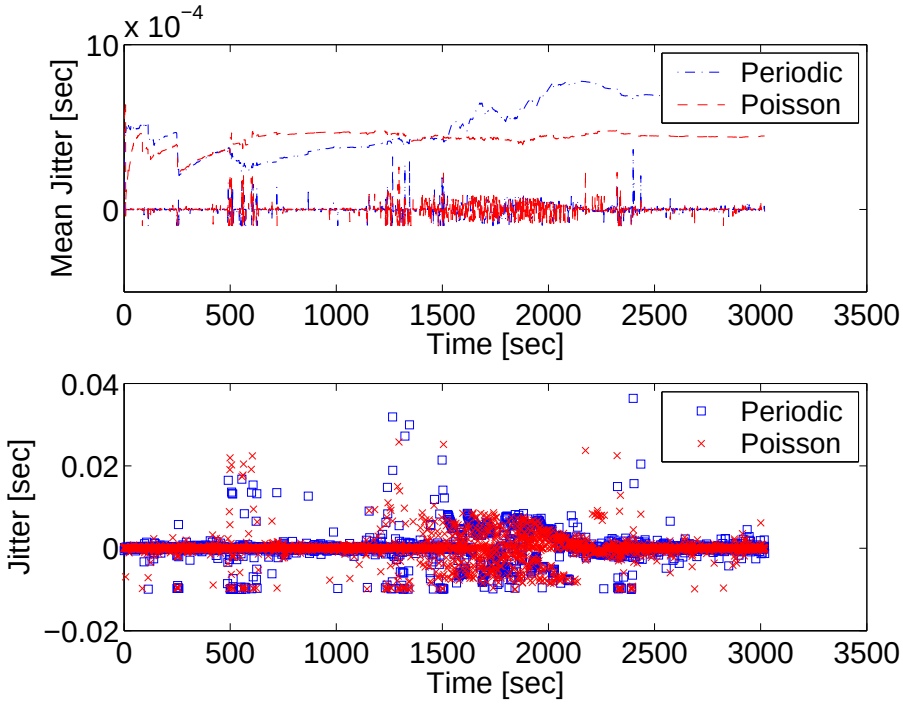

Figure 1.3. Difference in Packet Pair Delays ("Jitter") of 40-byte packets sent 10ms apart. (Top) Top set of curves are (staggered) long-term averages of the jitter while the lower set of curves are windowed averages of the last 100 jitter values. (Bottom) A scatter plot of the individual jitter samples.

strategies? PASTA guarantees asymptotic convergence, i.e., convergence in the limit. Given that most measurements consist of a finite number of probes which may be only a handful, is asymptotic convergence the goal of a sampling strategy?

- Assuming a measure of sampling performance, what is the optimal sampling strategy? PASTA has been used to justify Poisson probing, i.e., exponentially-distributed inter-arrival times. The ease of generating probe packets at fixed inter-arrival times makes periodic probing a possible sampling strategy.

- Does the sampling strategy depend on the path being measured? The "memoryless" nature of Poisson probing has been thought to its strength. What memory do, say, probes sent at uniformly distributed inter-arrivals possess? What about periodic probes?

- Given the desirability of keeping measurements as non-intrusive as possible, how do the answers to the above questions change?

- PASTA has been applied to observe delays and losses of individual packets[Pax99, ZDP01], pairs[SKK03], trains [LRLL04], and TCP throughput measurements [Pax99]. Can PASTA be applied to observe any property? If not, when is PASTA applicable? What sampling strategies should be followed for the other properties?

1.3.2 Inversion

Inversion is the question of "How do we obtain the desired property from the observed property?" Broad issues related to inversion that are of interest to us are given below. As before, we start with the basics and move to the more detailed.

- We discussed how, due to the non-zero size of probe packets, all active measurements are intrusive. What are the implications of this when the desired and observed path properties are the same, e.g., end-to-end delay? Can inversion be performed to remove the effect of probe packets?

- If the desired and observed properties are different, inversion is necessary. Such inversion is performed, for example, by measurements that attempt to access available bandwidth using probe delays [SKK03, JD03]. To perform such inversion, it is necessary to assume a particular system model. Most existing techniques use a single-hop model with fluid cross-traffic [JD04]. Given the discrepancy between fluid and packet-based cross-traffic [LRLL04], how do we model packet-based cross-traffic?

- Cross-traffic can be considered to be the only time-varying input to a unicast path. It influences delays, jitter, available bandwidth, etc. How do we access cross-traffic properties using active measurements? In fact, what properties can we even hope to access using fundamentally non-intrusive measurements? For instance, existing cross-traffic estimators [SKK03] access the mean rate of cross-traffic using close intra-pair gaps. Can we access more, say, the entire CDF?

- Earlier, we discussed the trade-off between non-intrusiveness and accuracy. In particular,

existing cross-traffic estimation techniques [SKK03] require very fine-grained control over probe timings. Is this possible in practice? Can this be relaxed?

- What are the practical implications of the answers to the above questions? For instance, what intra-pair separation can be used and on what paths?

1.3.3 Architecture Design

In the course of investigating the above questions related to sampling and inversion we find that it is very challenging to deal with the multi-hop nature of Internet paths especially using only non-intrusive measurements. This finding motivates the following questions.

- Measurements of what path properties are complicated by the multi-hop nature of paths? Clearly, end-to-end delay is easy to measure. What about per-hop delays [Dow99]?
- Are the multi-hop nature of paths and non-intrusiveness fundamentally impossible to tackle? If yes, why? If not, can we use newer primitives to tackle these challenges?
- How can the newer primitives be used to implement Measurement-Friendly Networks (MFNs), i.e., networks amenable to non-intrusive active measurements? If so, in what way is the measurement-friendliness achieved? What path properties can be measured in such networks? What techniques can be used to perform such measurements?
- What are the practical deployment issues with such an architecture? Is it easy to deploy them in a single ISP? What about end-to-end?
- Are there any fundamental limitations of active measurements that cannot simply be overcome even in MFNs? Do they impact accuracy? By how much?

1.4 Contributions

In this dissertation, we answer the above questions as broadly as possible. In some cases, we arrive at answers that are fairly general. In others, we use specific problems to provide new

insights. Our primary focus is on using observations of end-to-end delay. Though many of our results are likely applicable when the observed property is loss, we do not concentrate on this case. Our contributions are threefold. First, we investigate sampling-related questions and the role of PASTA in active measurements. We find that PASTA ignores variance as well as the need to remove the impact of probe packets. We formulate more relevant sampling strategies that can outperform Poisson probing. Second, we develop a fairly general FIFO-based model that is useful to develop inversion techniques to estimate bandwidth-related metrics. We use this model to explain existing techniques and to investigate the problem of cross-traffic estimation for the single-hop case in detail. We develop in-principle and practical techniques for the latter as well as motivate the hardness of extending these to analyze multi-hop paths. Third, we use network architecture to understand the potential and fundamental limits of non-intrusive active measurements especially on multi-hop paths. Specifically, we design a novel measurement-friendly network that makes it very easy to measure a variety of per-hop and per-path metrics. We now describe our contributions in more detail. Figure 1.4 shows how this dissertation progresses along with the main problems and results of each stage.

1.4.1 Sampling: The Role of PASTA

Traditionally [IPP, Pax99], the PASTA principle has been used to justify that sending probe packets, pairs and trains at Poisson epochs is necessary as a way to obtain unbiased estimates. We investigate whether this is justified or not assuming that inversion is only required to remove the impact of probe packets on the system and, hence, our observations. We use the theory of Palm calculus [BB03] to conduct our investigations in a rigorous manner. Palm calculus is ideal because it provides a framework to study issues related to sampling queueing systems, e.g., when are "event averages" (what a probe sees) equal to "time averages" (the underlying ground truth).

Using Palm calculus, we model the active measurement setting as a stochastic (random) system whose evolution depends on the input processes of cross-traffic and probing packets. This model is quite general and includes multi-hop systems and a variety of queueing disciplines. The power of this approach is evident once we realize that the exact nature of queueing (whether it is FIFO,

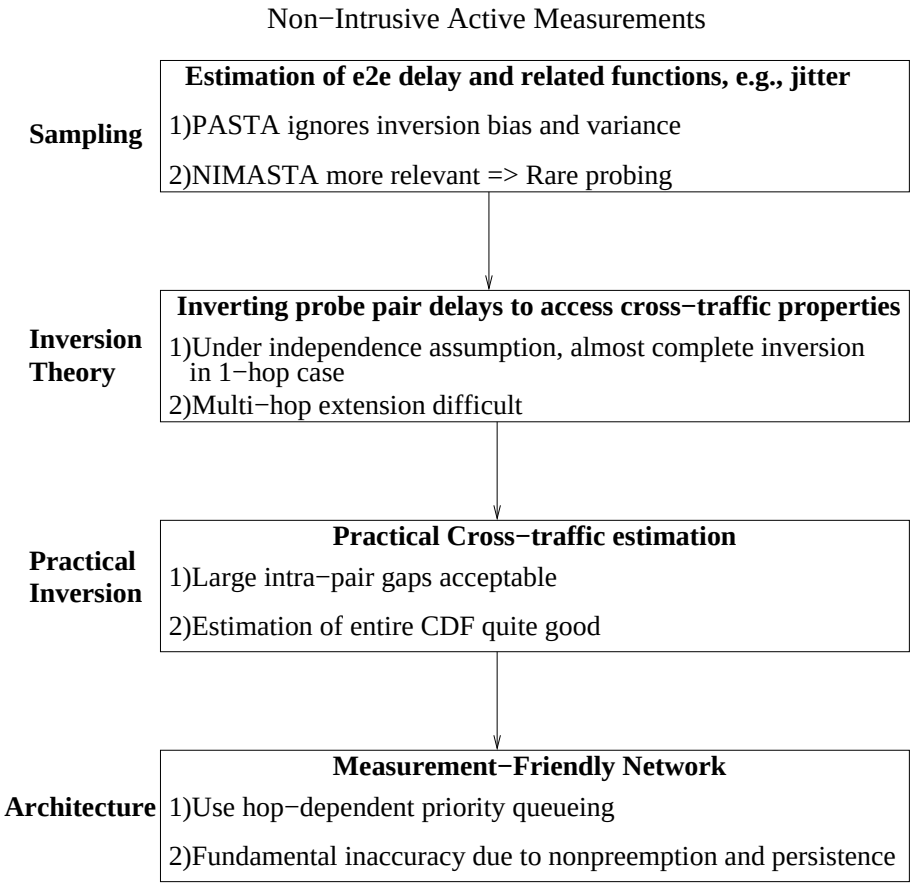

Figure 1.4. A flow-chart representing the progression of this dissertation. We show the four stages, the main problem we address, how we address it and our main results.

fair queueing, etc.) is irrelevant. What is required is only that the system (in particular, the desired property) be a deterministic function of the inputs.

In the above setting, we first consider completely non-intrusive probing achieved by sending zero-sized probes. Clearly, this is not a practically feasible situation. However, doing so allows us to distinguish the basic issues affecting sampling from the effects of intrusiveness on sampling. We find that, in the zero-sized probe case, the relevant principle is *NIMASTA (Non-Intrusive Mixing Arrivals See Time Averages)*. This principle states that zero-sized probes see unbiased estimates as long as they are sent at so-called *mixing* epochs. The mixing property is not unique to Poisson processes but shared by many others including any independent and identically distributed random process. For example, probing can be optimized to minimize estimation variance.

Then, we consider probes of non-zero size in which case NIMASTA no longer holds and the

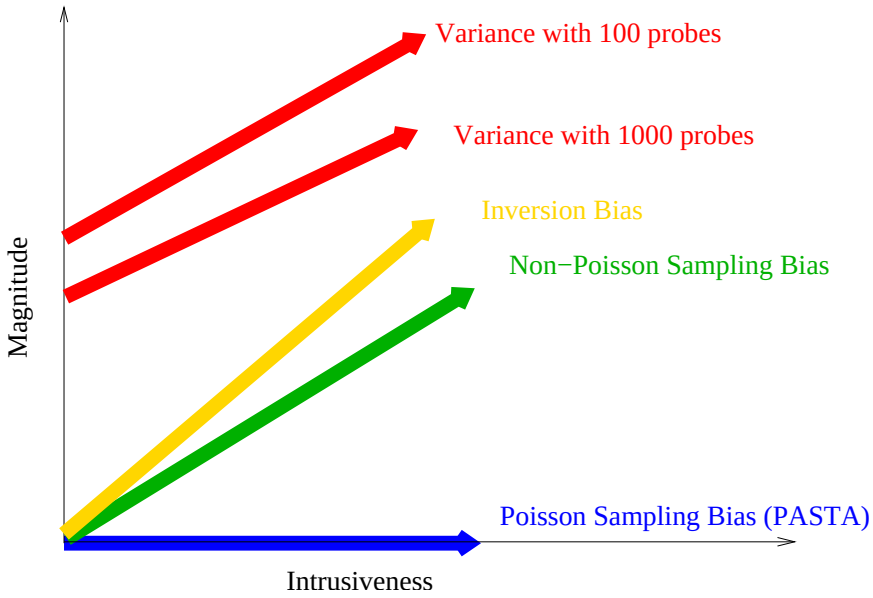

Figure 1.5. We plot the trends of the biases and variance, all of which together decide the optimality of a particular probing stream. All of them increase with intrusiveness, in general. PASTA says that sampling bias of only Poisson streams is zero. Other (mixing) probing streams have zero sampling bias when probes are of zero size. Variance typically decreases with increasing number of probes, for the same intrusiveness level. Note that the trends are not relative to each other. For instance, variance may be smaller than the inversion bias.

PASTA principle becomes valid. Intuitively, this is because probe packets perturb the system and hence, only a "memoryless" probing process such as Poisson would be "unbiased towards itself". But, the intrusive case is of little interest because intrusive probes only measure the perturbed system which includes the probes. Unbiased estimates of this system's properties are of little use in the absence of suitable inversion techniques that recover the unperturbed system properties. Moreover, to our knowledge, there is no known way to perform such inversion. Hence, simply using PASTA without inversion is of limited use. The only alternative (known) is to choose a "sweet spot" by sending probes at a rate that is low enough that inversion is not required for practical purposes but high enough that we can obtain good estimates within an acceptable time frame. Thus, rare probing is the best possible strategy. We briefly discuss how rare probing should be in practice.

The only advantage Poisson probes have over mixing probes is the lack of memory about themselves. This advantage is removed when, to eliminate the need for inversion, probes are made sufficiently rare. Overall, a probing stream suffers from bias due to lack of inversion and sampling (which is zero for Poisson). Estimation error is also caused by variance. Thus, whether to use Pois-

son or some other probing process depends on both kinds of biases and variance. A suitable probing process need not be Poisson. In fact, we show via simulations that this is indeed the case. Figure 1.5 presents a simplified picture of the various biases and variance and their relation to intrusiveness.

We also show that there is no basis for sending packet pairs, trains (not individual probe packets) or conducting throughput measurements at Poisson epochs. Rather, in all such cases, rare probing at mixing epochs can be used as an approximation to non-intrusive measurements. How rare probing should be and whether a particular mixing process is optimal is likely to be dependent on the cross-traffic characteristics. Our treatment also shows why non-mixing probe streams, e.g., periodic probes, often suffice in practice. We illustrate all these concepts and results using simulations. We could not use real Internet experiments since that would require us to calculate "time averages" which is impossible without very detailed models of routers.

In summary, we show that, contrary to conventional wisdom, diligent use of Poisson probing is not necessary. However, our work also indicates that determining the optimal probe stream is extremely hard and likely dependent on the system being measured. Hence, in practice, one of many reasonably good probing streams can be used.

1.4.2 Inversion: Cross-traffic Estimation with Packet Pair Delays

As discussed above, rare probing eliminates the need for inversion when the observed and desired properties are the same. Otherwise, inversion is often required. In general, a wide variety of methods can be used for inversion. Exploring all of them is not possible. Instead, we use our example problem to investigate the potential of probe pairs in accessing cross-traffic properties. We make four specific contributions. First, our work illustrates how to tackle the inversion problem: choose an appropriate setting, perform in-principle analysis that not only identifies what can be done, but also what *cannot* be done. Then, adapt the developed theory to practice. Second, we introduce a discrete-time FIFO-based framework that is well-suited to answer inversion questions in data networks especially as compared to fluid-based models used in prior work [HS03]. Third, we use this framework to introduce how we can perform identifiability analysis, i.e., what properties we can hope to invert or not, and develop in-principle inversion methods for cross-traffic estimation

in the single-hop case. We also motivate why it is hard to perform non-intrusive measurements on a path with multiple hops. Fourth, we demonstrate the power of non-intrusive active measurements by designing robust practical estimators of cross-traffic under fairly general conditions.

In-Principle Inversion

We consider a one-hop First-In First-Out (FIFO) path which is a reasonable approximation [HVPD04, SKK03] of typical network paths. We use Lindley's equation [BB03] from elementary queueing theory to relate the delays of two probe packets with the traffic that arrived to the hop between them. This relation shows that two functionals of cross-traffic are exposed by packet pair delays. These two functionals represent the average amount and "burstiness" of cross-traffic and are the only properties that we can *hope* to obtain. We then show that, in theory, almost the entire joint distribution of these two functionals can be determined under a reasonable assumption on cross-traffic. The portion of the joint distribution that cannot be determined depends on the size of the probes and reflects the cost of intrusiveness. We use a novel and intuitive geometric framework to explain how such inversion can be achieved once we recognize that certain conditional probabilities involving the delays of the pair form right angles in the space defined by the two functionals (see Figure 1.6). Furthermore, certain portions of the joint distribution that are dependent on the probe sizes are hidden. We also show that, without any assumption, the system is ambiguous, i.e., cross-traffic processes with different joint distributions can give rise to the same packet pair delays. Moreover, extending this theory to paths with just two hops is hard even under our assumption. We also show how our estimators and prior methods [JD03, HS03, SKK03] can be made more robust, by trading off non-intrusiveness, using our analysis framework.

Practical Inversion

Based on the theory we develop, we design practical estimators of average cross-traffic in the time between a packet pair and its joint distribution with the other functional representing the "burstiness" of cross-traffic. We face difficult issues related to estimating CDFs, e.g., monotonicity. We explore multiple ways in which these issues can be overcome. Using a combination of simu-

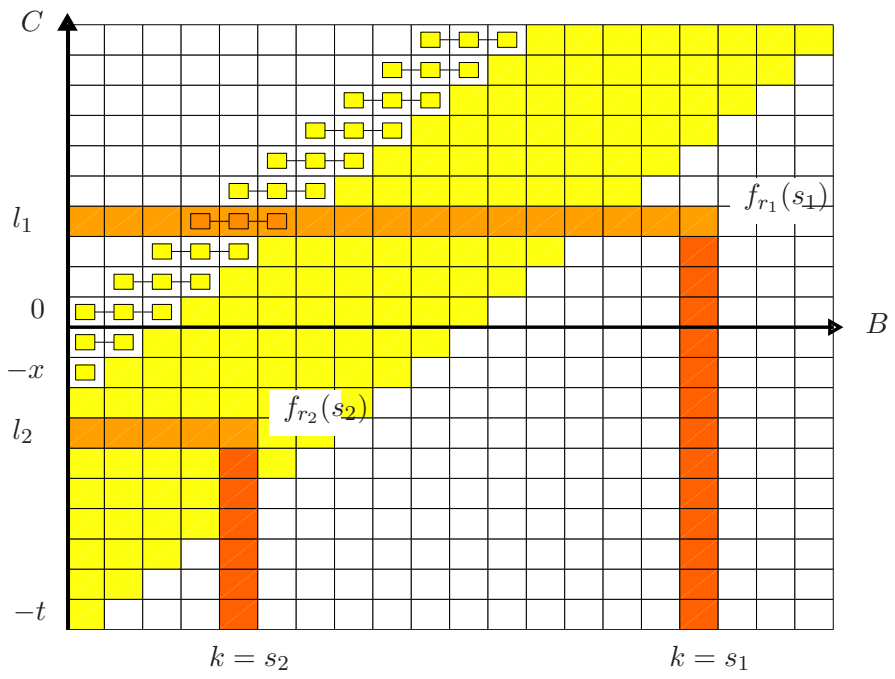

Figure 1.6. Our investigation into the in-principle potential of packet pair methods shows that two functionals of cross-traffic (B and C) are exposed by such delays. Their joint distribution is non-zero in a strip of size that is equal to the separation between the pair t. A sub-strip, proportional to the probe size x, cannot be inverted. The rest of the probability distribution can be inverted under a simplifying assumption. This inversion is achieved by noting that conditional probabilities involving packet pair delays form right angles in this space (as shown).

lations, passively-collected router traces and novel active experiments, we thoroughly benchmark our estimators. For hops with utilization higher than 50%, we can estimate the entire distribution of cross-traffic with an Mean Squared Root Error of less than 0.2. Moreover, this can be done with much larger intra-pair separation times than previous methods [SKK03] which makes them more robust to noise.

1.4.3 Architecture: Measurement-Friendly Networks

The hardness of measuring multi-hop paths non-intrusively motivates us to understand whether non-intrusiveness and the multi-hop nature can be simultaneously addressed. We turn to designing new network primitives that would allow these twin challenges to be tackled. The novel network primitives that we design are based on hop-dependent priority queueing. Specifically, we show that probe packets that have high-priority (with respect to normal data packets) at all hops except

a target hop essentially encounter queueing only at the target hop. This hop isolation property allows individual hops of a multi-hop path to be measured. We also show that low-priority (with respect to normal data packets) probes, by being transmitted only when there are no normal data packets, intrinsically measure the queue of the normal traffic. Hence, we explore the design of a *Measurement-Friendly Network (MFN)* using hop-dependent high and low priority queueing for probe packets.

In our proposed MFN, we design a variety of active techniques to measure end-to-end **and** per-hop properties such as delays, losses, cross-traffic, capacities and available bandwidth. Using simple simulations, we show that these are quite accurate. However, we find that fundamental inaccuracies arise, even in our MFN, due to nonpreemptive nature of packet forwarding as well as cross-traffic persistence. We find that existing Differentiated Services [BBC+98] functionality can be leveraged to deploy our architecture with minimal overhead within a single ISP. However, issues related to allowing end-users access to high-priority forwarding makes certain aspects of our architecture less likely to be deployed end-to-end. We discuss appropriate access control mechanism as possible solutions. We do not face any such issues if end-users are allowed to use hop-dependent low priority queueing.

1.5 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, we discuss prior work related to our dissertation. We survey work on active network measurements. In the context of sampling, our survey shows the pervasive use of PASTA for probe packets, pairs and trains without much insight into the various factors that decide the optimal sampling strategy. In the context of inversion, we show how existing non-intrusive techniques to estimate cross-traffic invariably require very fine-grained timing control or assume very specific cross-traffic models, thereby limiting their applicability. We also discuss a few works that have investigated network primitives to improve measurement accuracy. We find that some of the primitives are limited in scope while the rest of them focus on using costly, high-overhead network primitives relying on passive data collection. We also discuss theo-

retical work in queueing that we use, in our analysis. In Chapter 3, we describe our methodology and the tools used in this dissertation. Chapters 4-7 form the core of this dissertation.

In Chapter 4, we examine sampling-related questions that arise in the context of active measurements. Our starting point is the often-used PASTA principle. First, we use simple illustrative experiments to motivate why PASTA may be of limited relevance. Then, we use Palm calculus [BB03] to derive rigorous results that provide insights into the empirically observed behavior. We propose *rare probing* as a sound practical strategy.

In Chapter 5, we consider a simple one-hop path and two probe packets sent across this hop. We develop a simple packet-based model to describe the operation of this queue and in particular, relate the delays of the pair of probe packets. We find that two functionals of cross-traffic can possibly be exposed by inverting probe pair delays. Then, we show that, assuming that the delay of the first probe is independent of the cross-traffic between the pair, almost the entire joint distribution of these functionals can indeed be obtained. We also demonstrate two negative results. The first is without the assumption on cross-traffic, fine-grained timing control is essential for non-intrusive inversion. The second is our single-hop analysis cannot be extended to multiple hops.

In Chapter 6, we use our in-principle inversion technique for cross-traffic to develop practical estimators. We encounter practical estimation issues, notably, enforcing the monotonicity of estimated CDFs (cumulative distribution functions). We test our estimators using simulations, router traces and live experiments. We find that, for utilization as low as 50%, we estimated CDFs with (root mean square) error less than 0.2 using relatively large intra-pair separation times.

In Chapter 7, we investigate network primitives that guarantee non-intrusiveness and allow individual hops of a multi-hop path to be isolated. We find that it is possible to design a measurement-friendly network though cross-traffic persistence and the absence of preemption cause some inaccuracy that cannot be avoided. We present our conclusions and open questions in Chapter 8.

Chapter 2

Related Work

"We need to haunt the house of history and listen anew to the ancestors' wisdom."

- Maya Angelou, American poet, author and civil rights activist.

In this chapter, we discuss prior work related to this dissertation. This includes work in active network measurements as well as in other areas that we leverage, e.g., queueing theory. We start with key background material on the history of active network measurements in Section 2.1. The rest of this chapter is organized like the dissertation. In Section 2.2, we survey how previously proposed active measurements addressed sampling-related questions. In particular, we show that there are a number of basic questions surrounding the role of the "Poisson Arrivals See Time Averages (PASTA)" property in active measurements. In Section 2.3 we discuss the inversion techniques that prior works have used and motivate our work in Chapters 5 and 6 by showing that existing techniques are unsuitable for practical, non-intrusive cross-traffic estimation. In Section 2.4, we discuss prior work on architectural proposals to improve the accuracy of active measurements. We show that all of them inherently rely on passive data collection mechanisms. Moreover, no prior work provides insights into the fundamental limitations of active measurements, i.e., how accurate can active measurements be without such data collection mechanisms and why. We summarize in Section 2.5.

2.1 A Brief History of Active Network Measurements

Network measurements have played a key role in the evolution of the Internet. Some of the earliest and most widely-used measurement tools leverage(d) the ICMP (Internet Control Message Protocol) error-reporting functionality of IP networks. ICMP messages are generated, for example, when the Time-to-Live (TTL) of an IP packet reaches zero, a particular port on a host is unreachable or a request to generate a "host-alive" message is sent. Mike Muuss's *ping* [Muu83] program was one of the earliest tools to use ICMP. Using ICMP Echo requests, it measures the round-trip time to a destination host. Van Jacobson's *traceroute* [Jac87] is another widely-used tool that sends a stream of packets with increasing TTL values to a destination host. ICMP replies are generated at various hops from the source to destination. This enables users of the tool to determine the route from the source to destination and also, rough estimates of the round-trip times to each of the intermediate hops.

The proliferation of the Internet led to many studies of end-to-end Internet properties such as delay and loss. Often, these did not use *ping*. Instead, they relied on advanced tools that sent UDP and TCP packets of different sizes at specified times. Bolot [Bol93] conducted one of the earliest such studies using small-sized UDP packets sent at periodic time intervals. To perform it, Bolot used the *NetDyn* [SB93] tool. *NetDyn* had been developed initially to debug packet forwarding problems [SGA93]. Paxson [Pax97a, Pax97b, Pax99] performed a large-scale study of routing and packet dynamics by deploying a *Network Probe Daemon* on various sites of the Internet. In [Pax97a], he used *traceroute* to measure routing pathologies, stability and other miscellaneous routing characteristics. In [Pax99], Paxson used a similar experimental setup along with 100 Kilobyte TCP transfers (instead of *traceroute*). These transfers were then analyzed to study packet forwarding dynamics. Since then, many other techniques have been proposed to measure delay, loss and various other metrics such as capacity and available bandwidth. We discuss these in detail in Section 2.3.

Recognizing the importance of active measurements, many online websites provide access to measurement data, tools and sometimes, even allow users to conduct live measurements. CAIDA [CAI] provides many measurement tools, analysis engines and the results of analysis too. *Traceroute.org* [Tra] allows users to conduct *traceroutes* from any of their servers, located in many coun-

tries, to any target destination address. The Measurement and Network Analysis (MNA) arm of NLANR [MNA] also provides many traces and on-demand access-controlled measurement ability. The Internet End-to-End Performance Monitoring (IEPM) group at Stanford [IEP] provides similar functionality for sites involved in high-energy physics research. The National Internet Measurement Infrastructure (NIMI) [PMAM98, PAM00] initiative created a set of tools and system architecture for deploying a distributed measurement infrastructure in which a diverse set of users can conduct experiments in a secure manner. The Secure and Accountable Measurement Infrastructure (SAMI) [SAM] is the successor to NIMI. Recently, a similar infrastructure, Planetlab [Pla] has also become popular to perform wide-area experiments.

The importance of network measurements has also led to significant standardization efforts at the Internet Engineering Task Force (IETF). Specifically, the IP Performance Metrics (IPPM) Group at the IETF [IPP] is devoted to defining suitable performance metrics for Internet measurements and good rule-of-thumb principles on how to collect such metrics. Currently, they have standardized metrics such as one-way delay, connectivity, one-way loss, round-trip delay and packet delay variation.

2.2 Sampling

In this section, we review prior work on sampling-related issues of active network measurements, i.e., when to send probe packets. We first identify how this has been addressed in the networking community. Then, we review recent works that have questioned whether the current state-of-art is justified or not. Then, we review results from other fields, notably queueing theory, from where such results have been borrowed.

2.2.1 PASTA

Early active measurements did not investigate the impact of sending probe packets at specified times. The ease of sending periodic streams made it a natural choice. Hence, programs such as *ping*

used, as do their modern versions, periodic probe streams. Bolot [Bol93] also used periodic UDP streams to study wide-area delay and loss.

Paxson [Pax97a, Pax99] was the first to explicitly address the issue of when probe packets should be sent. Paxson applied the "Poisson Arrivals See Time Averages (PASTA)" principle of classical queueing theory [Wol82] in the networking context. He deduced that *any* active measurement is best performed at Poisson epochs as a way to obtain unbiased estimates. In [Pax97a], Paxson carried out measurements of the "routing state" in the Internet using *traceroute*s. Applying PASTA, he sent these *traceroute*s so that the time intervals between consecutive measurements of the same virtual path were exponentially distributed. In [Pax99], he applied PASTA to initiate 100 Kilobyte TCP bulk transfers by initiating them at exponential intervals. These bulk transfers were used to study pathological conditions such as packet reordering as well as characterizing loss and link capacities.

The IP Performance Metrics (IPPM) Group at the IETF [IPP] built upon PASTA and Paxson's results to recommend the use of Poisson sampling, for example, in RFC 2330 [PAMM98]. RFC 2330 also observes that non-Poisson probes such as uniform, geometric, additive random, or other probes can be used for a variety of practical reasons; for example, the interval between Poisson probes can be arbitrarily large or small, and such probes cannot be implemented in real systems. Hence, there might be a need to use implementable and "close enough to Poisson" probes, e.g., truncated Poisson probes.

Since then, Poisson probing has become part of the conventional wisdom of network measurements. In studying the constancy of delay, loss and throughput, Zhang, et al. [ZDP01] followed the same methodology used by Paxson in the earlier studies mentioned above. Interestingly, they performed TCP bulk transfers at periodic intervals rather than exponentially distributed intervals. To perform their analysis of losses and delays, however, they used Poisson probing. Wang, et al. [WZJ03] measured round-trip times on various Internet paths by sending trains of probe packets. They used PASTA to justify the sending of these trains at Poisson epochs.

Studies on bandwidth estimation have also relied on PASTA to justify sending packet pairs and trains at Poisson epochs. Strauss, et al. [SKK03] recommend the sending of packet pairs at Poisson

epochs to obtain unbiased estimates. They use these estimates to infer the mean cross-traffic rate and available bandwidth. In rigorously analyzing available bandwidth estimation techniques based on packet trains, Liu, et al. [LRLL04] assumed that packet trains sent at Poisson epochs guarantee unbiased sampling. Other bandwidth estimation studies have ignored PASTA and used periodic probe trains. Jain, et al. [JD03] sent periodic probe trains as part of their *pathload* tool to measure available bandwidth. The individual trains themselves were periodic.

In Chapter 4, we investigate if the extensive extensive use of PASTA in active network measurements, as detailed above, is justified. We find that, for most practical purposes, PASTA is irrelevant because it does not offer any guidance on how the effect of probe packets can be eliminated. By considering the resulting bias and estimation variance, we find that non-Poisson probing streams can be better than Poisson probing streams. We also show that using PASTA to justify sending packet pairs or trains at Poisson epochs (for example, see [SKK03]) is not justified.

2.2.2 Beyond PASTA

Recent work has attempted to better understand the impact of PASTA and the design of estimators for active probing. Bin Tariq, et al. [TDDA05] empirically examine the difference between Poisson and periodic sampling, and show that, in many cases, the difference between estimates of delay and loss obtained with Poisson and periodic probes is not significant. They also state that PASTA may not apply to multi-hop paths since Poisson probes may not arrive as Poisson streams at bottleneck links. It is not clear if they refer to measuring per-hop metrics or end-to-end metrics. Our work in Chapter 4 provides the theoretical framework to clarify and rigorously understand their observations.

Sommers, et al. [SBDR05] set out to understand the probing process best suited to measure packet loss. They propose the use of a geometrically distributed packet pair (or packet triple) to estimate the duration of loss periods. They find that this performs better than Poisson probes. Their analysis is conducted in a discrete-time setting. Note that the geometric distribution is the discrete-time equivalent of exponential distribution.

Roughan [Rou05] analyzed Poisson probing as a way to understand the impact of correlations in

the observed (delay) process. For analytical tractability, he used a single M/M/1 queue, i.e., Poisson cross-traffic arrivals with packet sizes distributed exponentially too. He used this simple one-hop case to show that such correlations make it less attractive to increase probing rates, and negatively impact estimator variance. Roughan's work is similar in spirit to our work in Chapter 4 in that it is another step towards developing estimators that take not just bias but also variance into account.

2.2.3 On PASTA-like results

We now provide a brief overview of prior works that derived PASTA and related properties. The problem of identifying the conditions under which observations of a stochastic system coincide with the stationary distribution of the observed process has a long history, starting with Descloux in 1967 [Des67]. Wolff named, gave the first rigorous proof for, and popularized the PASTA principle [Wol82], although that principle had been investigated in prior works in less general settings [Bru71]. Wolff considered any stochastic process which arrivals observed *and* interacted with. He showed that the fraction of such arrivals that observe a particular state, e.g., number of customers in a queueing system, asymptotically converges to the fraction of time the stochastic process is in that state if the arrival process is Poisson.

Melamed and Whitt [MW90] later derived conditions for ASTA ("Arrivals that See Time Averages") to hold: first, a sufficient condition, which is essentially a weaker version of the Lack of Anticipation Assumption used by Wolff, then a necessary and sufficient condition for stationary settings. Melamed and Whitt [MW90] also derived theoretical expressions quantifying the amount of bias as a function of the property being measured and the arrival process. A more detailed survey of the ASTA-related queueing theory work can be found in [MY95]. Palm martingale calculus [BB03] is a modern queueing theory framework that is especially useful to understand when event averages equal time averages. The PASTA property is also derived in this context. We use this theory of Palm martingale calculus in Chapter 4 to investigate the role of PASTA in active network measurements.

2.3 Inversion

In this section, we survey the inversion-related contributions of prior works. These are the background material for our work on theoretical and practical inversion in Chapters 5 and 6. We start with a discussion on the packet spacing effect on store-and-forward links that is the basis of most inversion techniques. Then, we survey capacity and available bandwidth - two specific inversion problems that have received a lot of research attention. We end by discussing inversion techniques to access other metrics too.

2.3.1 Packet Spacing Effect

One of the earliest requirements for active measurements arose in the context of congestion control. In his seminal work, Jacobson [Jac88] laid the foundations of many of the mechanisms found in TCP today. This was made necessary after the Internet, which was still in its infancy, suffered a series of congestion collapses in 1986. This was diagnosed [Jac88] as being due to the lack of good mechanisms to determine the rate at which packets should be sent into the network. Jacobson used a schematic of a single slow link (representing the *bottleneck*) to illustrate that the packets of a sender reach the receiver with a spacing that depends on the bottleneck bandwidth. Arguing that this spacing is preserved on other higher-speed links, Jacobson motivated the "self-clocking" mechanism as a good way to ensure that congestion collapses are prevented. His analysis is simplistic because it ignored cross-traffic packets which may be increase/decrease the inter-packet spacing. However, the self-clocking mechanism itself is valid and is used in TCP even today.

Keshav [Kes95] analyzed the bottleneck spacing effect in more detail *assuming* a round-robin scheduling discipline. Keshav proposed packet-pair based flow control schemes based on this effect. The basic idea behind this approach was to use packet pairs to estimate the *available bandwidth* to the flow and send data at that rate. The rate was continuously monitored and hence, dynamically updated. Rate-based congestion control mechanisms such as NETBLT [CLZ87] also set the sending rate (over a window of packets) to the spare capacity along the path.

The packet-pair analysis of Keshav [Kes95] is not applicable for most Internet paths, then and now, because most links use the First-In First-Out (FIFO) scheduling discipline. Bolot [Bol93]

applied rigorous queueing theory principles to study the inter-packet spacing on such FIFO links. He observed that cross-traffic packets often cause the inter-probe spacing to be different from the bottleneck bandwidth. However, a careful analysis showed that the bottleneck spacing can indeed be extracted. Bolot used the Lindley's equation [Kle75a, BB03] in this analysis. Later, we also use this to understand the in-principle potential of active measurements and to design practical cross-traffic estimators.

These initial works on packet pairs have led to two kinds of techniques - those that estimate link capacities and those that estimate available bandwidth. We discuss the former first.

2.3.2 Capacity Estimation

There are two kinds of capacity estimation tools. The first kind use probe packet delays. This was pioneered by Van Jacobson in a *traceroute*-like tool called *pathchar* [Pat97]. He proposed that packets of different sizes that expire at the same hop be sent. Return ICMP replies from them are used to determine the minimum delay taken by each size. If these are the true minimums, the delay would increase linearly. Linear fitting is used to determine per-hop propagation delay and capacity. The true minimums are estimated by sending multiple probe packets of each size that expire at a hop. Downey [Dow99] performed extensive measurements with *pathchar*. His *clink* [Dow] and *pchar* [Pch] are different implementations of *pathchar*.

Lai, et al. [LB00] proposed *nettimer*, which used a large *tailgating* packet to significantly reduce the load of *pathchar*-like methods. They also eliminated the requirement for ICMP replies at the cost of destination cooperation. Pásztor, et al. [PV02a] continued on this theme and proposed the use of *packet quartets* to minimize multi-hop queueing effects on capacity estimation.

A second kind of capacity estimation tools estimate only the smallest capacity along a path. Carter, et al. [CC96] proposed *bprobe* that used the packet pair dispersion as the narrow link capacity. However, queueing at other hops can change such dispersion. Hence, they improve the accuracy of their estimates by employing filtering algorithms that use dispersions seen by pairs of various sizes. Lai, et al. [LB99] used more sophisticated filtering algorithms for the same purpose. Paxson [Pax99] analyzed the bulk transfer traces he collected to infer bottleneck bandwidth. He ob-

served multiple modes of packet dispersion. Hence, he used *Packet Bunch Modes (PBM)* that used packet trains of different lengths to recognize pathological conditions such as multi-channel links. He also used heuristics to choose an appropriate dispersion value as the dispersion representing the narrow link capacity [Pax97b].

The multi-modal nature of dispersion was studied in more detail by Dovrolis, et al. [DRM01]. In particular, they coined the term *asymptotic dispersion rate (ADR)* as the most observed packet dispersion that is different from capacity. They implemented a heuristic-driven tool for capacity estimation, *pathrate*. Pásztor, et al. [PV02c] also performed an in-depth analysis of the effect of packet probe size on capacity estimation. Recently, Kapoor, et al. [KCL+04], proposed a filtering algorithm that picks out the smallest *combined* delay of the probe packet pair and uses it to calculate the capacity. The rationale is that the smallest combined delay is caused when both packets do not see any unexpected queueing except at the bottleneck hop. Their tool, based on this, is called *Capprobe*. Though based on sound principles and mostly accurate, these techniques can sometimes be negatively affected by queueing. In Chapter 7, we show that our proposed Measurement-Friendly Network primitives make these techniques much more robust to undesirable queueing effects. Capacity estimates obtained using them can be used in conjunction with our cross-traffic estimators in Chapter 6 to estimate available bandwidth. Next, we discuss techniques that estimate spare capacity along a path.

2.3.3 Bandwidth Estimation

We now discuss techniques that estimate bandwidth-related metrics that help determine the "spare capacity" along a path. Some of the earliest such techniques were congestion control techniques. Indeed, as we discussed previously, the packet spacing effect was first described in the context of Jacobson's work on TCP congestion control mechanisms [Jac88]. The onset of congestion, noticed when packets are dropped [Jac88], is one of the most widely used ways by which TCP data flows determine that they have exceeded the "spare capacity" on the link. Jain [Jai89] proposed the use of increasing delays to make the same determination. Brakmo, et al. [BOP94] adapted this principle to TCP and called it TCP Vegas. Given the importance of "spare capacity" in congestion control, Mathis and Allman [MA] developed a formal definition for this metric. They called it the

Bulk Transfer Capacity (BTC) and defined it as the sending rate determined by congestion control algorithms as the Allman, et al. [AP99] discussed received-side techniques to improve the efficacy of estimating BTC.

Paxson [Pax99] defined the so-called *relative available bandwidth* metric, β. This metric calculates, in a rough sense, if two consecutive packets of a probe flow had cross-traffic between them or not. This is used to determine the metric. If β is 1, then there is little cross-traffic and hence, utilization is close to 0. Otherwise, it is close to 1. Though it is well-defined, relative available bandwidth is hard to relate to available bandwidth. This led to techniques that specifically estimate *available bandwidth*.

Available bandwidth is defined as the maximum rate that an end-to-end flow can obtain without reducing the rates of existing flows [JD03]. This is different from BTC because the BTC definition assumes that the rates of existing flows are reduced upon probing. Carter, et al. [CC96] proposed using the dispersion of long packet trains to estimate available bandwidth. They built a tool, *cprobe*, that implemented their estimation technique. Dovrolis, et al. [DRM01] used extensive simulations to show that such dispersion is subject to not the available bandwidth but a complex function of hop capacities and cross-traffic that they termed asymptotic dispersion rate.

Problems with interpreting packet train dispersions have led to techniques that use the same principle as TCP Vegas [BOP94], i.e., increasing delays imply that the probing rate has saturated the path and hence, is equal to the available bandwidth. Jain, et al. [JD03] proposed a tool *pathload* that sends widely-spaced periodic trains. Using a parametric, heuristic-based algorithm *pathload* determines whether the delays of a packet train are increasing or not. *Pathload* estimates the available bandwidth range by iteratively sending trains with various probing rates. The tool, TOPP, proposed by Melander, et al. [MBG00] is very similar. TOPP sends a train consisting of a pair of packets. Whether or not delays are increasing is determined by observing the delays of the pairs of packets. Pathchirp [RRB$^+$03] is very similar, too. It sends trains in which the spacing between packets of a train reduces exponentially. Heuristic algorithms are used to invert these delays to obtain available bandwidth. Finally, Hu, et al. [HS03] proposed IGI and PTR, two similar techniques that only look at the first and last packet of a probe train.

All of the above methods that look for increasing delays are fundamentally intrusive because they have to cause observable increases in queueing delays. Of course, by spacing out probe packet trains, these are made effectively non-intrusive. In contrast, our work in Chapters 5 and 6 do not rely on queue buildups. Nevertheless, in Chapter 6, we use our framework to describe how these prior methods can be made robust to false positives. All these prior works used a fluid-flow cross-traffic model of a single hop with a packet-based probing model. Recent work [LRLL04, LRL05b, LRL05a] showed that inaccuracies can arise out of using a fluid-flow cross-traffic model. Specifically, they showed that packet-based cross-traffic from non-bottleneck links on a path can add noise to delay measurements. These measurements can make available bandwidth measurements inaccurate. The likelihood of inaccuracies is reduced when the length of trains is increased. Thus, they show the tradeoff between effective non-intrusiveness and estimation accuracy. This tradeoff arises due to the fundamental intrusiveness of such techniques especially on paths on which more than one hop has significant queueing. Unlike us, they do not present any new inversion techniques.

A different set of techniques estimate available bandwidth by subtracting the estimated average cross-traffic rate from the capacity of the bandwidth-constrained link of the path. The capacity itself is determined using one of the methods we discussed earlier. Spruce [SKK03] uses the delays of closely-sent packet pairs to estimate average cross-traffic rate. The advantage of these methods is that they are fundamentally non-intrusive. The disadvantage is that not only are they single-hop but they also require very tightly controlled intra-pair spacing. Hence, they are not robust to queueing at non-bottleneck links. In Chapter 5, we develop inversion expressions that do not require such small intra-pair spacing. We evaluate practical estimators based on them in Chapter 6. Additionally, our estimators estimate not just the mean cross-traffic rate but also the entire distribution. Other prior works [ANT01, RCR$^+$00, SM98] focus on cross-traffic estimation specifically *assuming* a specific cross-traffic arrival process, e.g., multi-fractal wavelet, Poisson. Based on this assumption, they attempt to estimate the parameters describing these processes. In contrast to the parametric cross-traffic estimation methods, we make a well-justified assumption that is quite general in Chapter 5.

2.3.4 Other Examples of Inversion

Apart from capacity and available bandwidth estimation, inversion techniques have arisen in other contexts. Many prior works [Pax99, SBDR05, ZDP01] invert observed losses of probe packets to derive loss characteristics of the corresponding end-to-end path. Paxson [Pax99] was among the first to observe that droptail queueing leads to bursty, not independent, losses. Zhang, et al. [ZDP01] studied the time durations over which loss properties are unchanging. They considered loss episodes, their durations and loss rates. Recent work by Sommers, et al. [SBDR05] considered techniques that rely on multiple sets of packets to better estimate loss properties. Since we do not consider loss estimation in this dissertation, we refer the reader to [SBDR05] for a detailed review of loss estimation techniques.

The tool *pathchar* [Jac97, Pat97] not only inverts the capacities of individual hops but also the per-hop propagation delay. This is done by subtracting the delays of ICMP replies from consecutive hops. In doing so, *pathchar* assumes that the ICMP replies encounter negligible queueing. The minimum of these per-hop delays is considered to be the propagation delay. Downey [Dow99] considered inverting the observed ICMP round-trip times to obtain per-hop queueing delays. The delays from the ICMP replies provide a distribution of end-to-end delay until each hop of the path. Downey models the distributions as lognormal and deconvolves them. The "deconvolution difference" in the distributions of delay till consecutive hops is estimated to be the distribution of per-hop queueing delays. In Chapter 7, we show that this is not true because cross-traffic persistence can cause an unavoidable bias in estimating per-hop metrics.

Inversion has been used for multicast-based inferences too. For instance, Tsang, et al. [TYNB04] estimate delay variance on the shared segment from the probe source to two destinations. We do not review other such techniques since we primarily focus on unicast path estimation. Inversion techniques to estimate the scheduling discipline used have also been proposed. Kuzmanovic, et al. [KK01] use a single-hop model to reverse-engineer the queueing discipline in a QoS-enabled model and the parameters of the queueing discipline, e.g., weights of weighted fair queueing. Wei, et al. [WWZ$^+$05] proposed inversion techniques to classify the type of access links used at the destination. The field of perturbation analysis [Gla91] addresses the question of estimat-

ing the characteristics of an unperturbed queueing system from observations of a perturbed system. But, the questions they answer typically assume very specific models of the system and are not directly applicable to the Internet.

2.4 Architecture and Fundamental Limitations

As we discussed above, techniques have been proposed for estimating available bandwidth and cross-traffic properties. Many of them are fundamentally intrusive because they rely on observing increasing delay trends on saturating available bandwidth. Recent work by Liu, et al. [LRL05a] showed rigorously that the burstiness of cross-traffic at other hops of a multi-hop path can impact the accuracy of estimation. Little formal analysis has, however, been done on non-intrusive techniques that estimate cross-traffic properties, e.g., [SKK03]. The analysis in [LRL05a] is a step towards understanding the potential of active measurements. However, their analysis does not provide any insights into what *can* be measured. Moreover, they are focused on analyzing multiple input rates and derive asymptotic results. The potential of active measurement techniques is also faced with tools such as *pathchar* [Jac97, Pat97] that estimate per-hop properties such as queueing delays. Tariq, et al. [TDDA05], on the contrary, state that the lack of Poisson probing *to* a particular hop makes it hard to estimate per-hop properties. Thus, it is not clear whether or not per-hop properties can be estimated or not. In Chapter 7, we propose network primitives designed explicitly to make active measurements more powerful and accurate. We find that per-hop properties can be estimated in our Measurement-Friendly Network architecture except for unavoidable accuracies caused by nonpreemption and cross-traffic persistence.

Network architecture design has often be used to enable a network with better functionality. Accurate measurements are no exception. Seshan, et al. [SSK97] proposed SPAND, a network architecture that allows for multiple users to share information on end-to-end performance information. Varghese, et al. [VE03] showed how router measurements can be used to improve the efficacy of active measurements. They showed this by designing architectures for estimating traffic matrices and measuring route stability. They used per-prefix traffic counters, etc., in their architectures. The ICMP protocol was originally an error-reporting protocol. But it has been used extensively

in active measurements. Luckie, et al. [LMB01] explicitly designed a protocol for aiding active measurements. Their IP Measurement Protocol (IPMP) overcomes issues related to clock precision and drift, protocol-dependent forwarding and low overhead. To our knowledge, no prior architecture improves the efficacy of active measurements without passive data collection mechanisms. Our Measurement-Friendly Network architecture proposed in Chapter 7 shows that this is possible - we use hop-dependent priority queueing to better estimate per-hop and path properties non-intrusively.

2.5 Summary

We started this chapter by first discussing the historical evolution of network measurements. Then, we discussed how previous techniques have approached the sampling-related aspects. We discussed the conventional wisdom on using the PASTA principle as a motivation to send Poisson probe packets, pairs and trains. The discussed prior work on PASTA motivate us to better understand the role of PASTA in active measurements in Chapter 4. Then, we discussed a variety of known inversion techniques. We divided them into those that estimate capacity estimation, spare capacity and other metrics. In particular, we discussed prior work related to using probe packet pairs. We showed that prior cross-traffic estimation techniques either assume very specific parametric models of cross-traffic or require tightly controlled intra-pair gaps. Moreover, the latter only estimate the mean cross-traffic rate. Our work in Chapters 5 and 6 addresses these shortcomings by showing how the entire cross-traffic distribution can be estimated using relatively larger intra-pair gaps. Finally, we used prior work to demonstrate the effects of multi-hop queueing on the accuracy of active measurements. In addition, all architectural proposals to improve accuracy have directly or indirectly relied on accessing more data from routers. No prior work investigates whether similar levels of accuracy can be obtained from active measurements using network primitives that do *not* rely on router data. Consequently, operators of today's networks rarely rely on active measurements except for simple end-to-end delay, loss and router-level information. This motivates our work in Chapter 7 where we design novel forwarding primitives that make a network much more amenable to active measurements. In that chapter, we also show that unavoidable inaccuracies that arise in our

proposed architecture are often small or can be overcome. This, along with its ease of deployment, make our architecture a practical, low-cost alternative to passive measurement infrastructures.

Chapter 3

Methodology

"Let your intentions create your methods and not the other way around."

- Peter McWilliams, American author.

In this chapter, we describe our methodology and the tools we used or developed to perform our work. As discussed in Chapter 1, we investigate three aspects of active network measurements in this dissertation. These are related to sampling, i.e., when to send probe packets, inversion, i.e., using the observed delays to access the desired metrics and network architecture, i.e., network primitives to improve the accuracy of active measurements. For each of these aspects, we followed a three-pronged approach (see Figure 3.1). First, we used prior works to determine the current state-of-art and analyzed their shortcomings. Second, we used rigorous theory, simulations and modeling to understand how these shortcomings can be addressed. Based on this, we developed newer insights and techniques. Third, we used simulations and/or real experiments to fine-tune and validate practical techniques. There was also a feed-forward mechanism in our problem selection process, i.e., earlier parts of the dissertation influenced later parts. For instance, our investigation of network architectures in Chapter 7 is motivated by earlier chapters, in which we encountered challenges related to multi-hop queueing effects.

This chapter is organized as follows. In Section 3.1, we describe the theoretical concepts that we used in this dissertation. We describe elementary probability and statistics as well as advanced ideas from queueing theory. In Section 3.2, we describe the various simulation frameworks that we

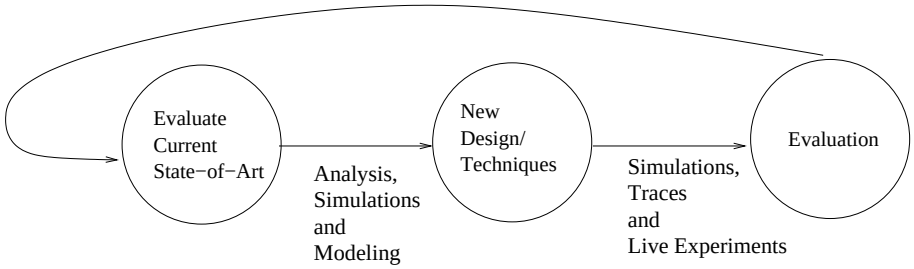

Figure 3.1. A schematic of the methodology followed. For our investigation into sampling-related aspects, inversion and network architecture, we first used prior work to motivate the target problem and the shortcomings of existing work. For instance, our investigation in Chapter 4 is motivated by the lack of prior work thoroughly investigating the role of PASTA. Earlier parts of the dissertation also influenced later parts. For instance, our investigation of network architectures in Chapter 7, to improve the accuracy of active measurements, is motivated by the challenges faced in Chapter 4 and 5 due to multi-hop queueing effects. Then, we used one or all of analysis, simulations and modeling to understand how these shortcomings can be addressed. Finally, we evaluated our new design/techniques. Our evaluation was performed using simulations, router traces and/or Internet experiments.

developed and used. These include existing simulators, modifications to them that were useful as well as simulators that we developed from scratch. We also describe the analytical tools used to process the data obtained via simulations. In Section 3.3, we describe the experimental setups we used to collect traces of real Internet cross-traffic. We summarize in Section 3.4.

3.1 Theoretical Concepts

In this dissertation, we repeatedly use three elementary concepts from probability theory and statistics [Dur96, Sto95]. We provide an overview of them now. The first concept is that of *independence* between random variables. Two random variables X and Y are defined to be independent if

$$P(X \in \mathcal{A}, Y \in \mathcal{B}) = P(X \in \mathcal{A})P(Y \in \mathcal{B}) \tag{3.1}$$

where $\mathcal{A}$ and $\mathcal{B}$ represent a subset of values that X and Y can respectively take. In addition, if X and Y are independent, then the following is true.

$$\begin{aligned} P(X \in \mathcal{A} | Y \in \mathcal{B}) &= \frac{P(X \in \mathcal{A}, Y \in \mathcal{B})}{P(Y \in \mathcal{B})} \\ &= \frac{P(X \in \mathcal{A}) P(Y \in \mathcal{B})}{P(Y \in \mathcal{B})} \\ &= P(X \in \mathcal{A}) \end{aligned} \tag{3.2}$$

The first reduction is by Bayes' theorem [Dur96] and the second by independence. We use conditional probabilities and Equation 3.2 in Chapters 5 and 6. The second concept that we use is related to that of estimation. Consider a random variable, X, that we are interested in estimating. Denote a candidate estimator as $\hat{X}$. Then, the *bias* of the estimator is $E[\hat{X} - X]$. Every sample $\hat{X} - X$ need not differ by the bias since the latter is only the expectation of the difference. In other words, the estimated quantity may exhibit *variance*. The third concept that we use is that of estimation error. As explained above, bias and variance represent two ways of quantifying error. A common way of combining both of them is to use the Mean Square Error (MSE) defined as the $E[(\hat{X} - X)^2]$ which is also equal to $bias^2 + variance$. The square root of MSE is called the Root Mean Square Error (RMSE). We use MSE and/or RMSE to quantify performance of estimators.

In Chapter 4, we use Palm Martingale Calculus [BB03] to study sampling-related questions. Palm calculus is ideal because it answers questions of interest to us, namely, those relating event averages and time averages in queueing theory. In the active measurement framework, the former represent what probing sees and the latter represent the underlying *ground truth*. Using suitable abstract frameworks and ergodic theory [Pet83], Palm calculus answers such questions.

3.2 Simulations

As always, simulations are a natural way to study the properties of a system using an appropriate model. The inputs and model can be controlled as necessary and any property can potentially be measured. This helps us understand the fundamental effects in play and to fine-tune parametric settings. We now describe two simulators that we used and a novel data analysis tool that we needed for our investigation.

3.2.1 *qsim*: A Queueing Simulator

The widely-accepted model of a network path is that it is a queueing system. The First-in First-out (FIFO) queueing discipline is used predominantly though some access networks use other disciplines [LPP04, WWZ+05]. The FIFO nature of widely-deployed routers is also validated in [HVPD04]. Hence, we built *qsim*, a tool that simulates a queue or a network of FIFO queues. Since *qsim* is not packet-based, it is computationally very efficient.

qsim leveraged the code of a more rudimentary queueing simulator *psim* [Pó1]. The inputs to *qsim* are a description of the propagation delays and capacities of various hops along a path. The cross-traffic arrival processes at each link are specified either using a trace-file or by specifying specific arrival processes, e.g., Pareto. The probing stream can also be similarly specified and flows from the first hop to the last hop. The output is configurable in the sense that the arrival and departure times of each packet stream (cross-traffic and probing) at every hop can either be captured or not.

qsim works in stages, each stage considering a single probe packet. At the end of each stage, the "hop time" is advanced to the arrival time of that stage's probe packet at that hop. This is done by considering, in order of arrival, cross-traffic packets. One variable is maintained for each hop, the time at which the queue (of that hop) would empty if there is no additional cross-traffic. This is used to calculate the queue size and departure time of the next cross-traffic packet. Suitable variables are updated and the process is continued for all cross-traffic packets. The output generated includes packet arrivals and departure times.

Our `C` implementation of *qsim* was done in about 2000 lines of code. We used a `Matlab` implementation of *qsim* for a single-hop path too.[1] Our implementations allowed cross-traffic inter-arrivals and packet sizes to be generated according to a variety of random processes including exponential, uniform, constant and Pareto distributions.

3.2.2 ns-2

To generate realistic traffic comprising of TCP flows, web traffic, etc., we also found it necessary to use a more functional simulator. Our choice was the well-known packet-level simulator, *ns-2*

[1] We owe thanks to Darryl Veitch for writing this code

[Sim]. *ns-2* is event-driven and implemented using `C++` and `OTcl`. The latter provides scripting functionality to easily invoke *ns-2*, the core of which is written in `C++`.

An *ns-2* simulation is invoked using an `OTcl` file that specifies the topology, the capacities and propagation delays of the various hops and the flows traversing them. Pre-defined flows include many variants of TCP, UDP-based flows (with inter-arrival times that belong to one of many distributions) and web traffic. Simulations can be configured to collect traces of packets traversing any hop.

The scheduling discipline used at each hop is FIFO, by default. In designing measurement-friendly networks, we required the use of priority queueing. Priority queueing is implemented in the Diffserv [BBC+98] modules of *ns-2*. These modules provide 4 queues at each hop, with traffic being classified into a queue based on its Diffserv codepoint [BBC+98]. The classification scheme at each hop can be different thereby allowing us to simulate hop-dependent priorities.

We also had to modify *ns-2* to accept packet sizes of zero. The modification was minor and required a few lines of code. This was required to study hypothetical active measurement scenarios involving probe packets of zero size. Such hypothetical scenarios were studied to understand the role of intrusiveness with respect to sampling bias in Chapter 4.

3.2.3 Ground Truth Calculator (GTC)

Often, we need to compute the "ground truth" about a network path property. For instance, the ground truth of end-to-end delay represents the time-averaged behavior in the following sense. Let $D(t)$ represent the end-to-end delay that a (hypothetical) single packet of size, say 100 bytes, would have experienced if sent at time t. The ground truth end-to-end delay is the function $D(t)$ over all time t. The mean ground truth end-to-end delay is the mean of this time-varying function.

In this dissertation, we find it necessary to calculate the ground truth of the inputs (cross-traffic) and derived path properties (e.g., end-to-end delay, jitter). The former is easily done since cross-traffic arrivals are known (we specify them) and the arrivals form a discrete process that is easily counted. However, accessing the ground truth about derived (continuous time) properties such as $D(t)$ required us to build a *Ground Truth Calculator (GTC)* described below.

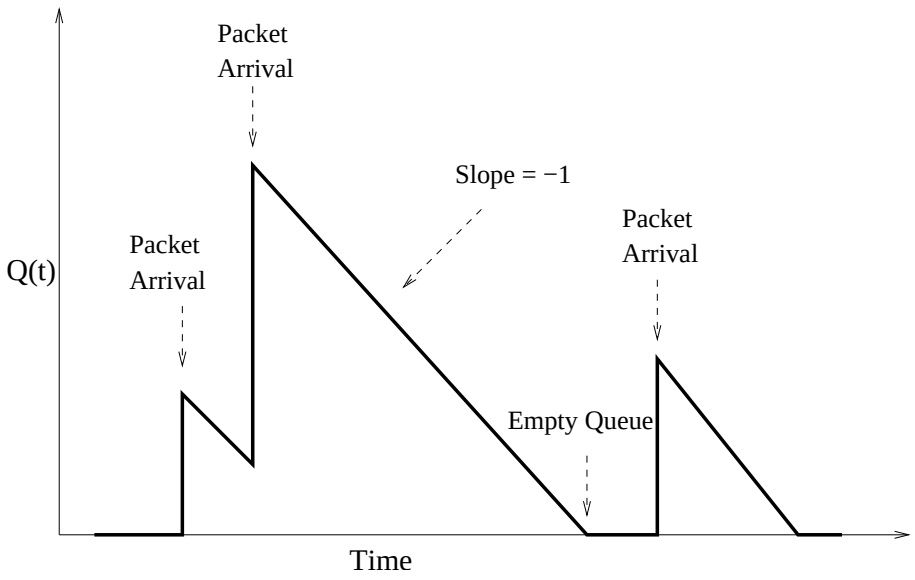

Figure 3.2. An illustration of the nature of $Q(t)$, the queue size in units of transmission time. $Q(t)$ is discontinuous when a new packet arrives. It has a slope of -1 when a packet is being transmitted and is horizontal when the queue is empty.

We first describe how $D(t)$ can be calculated. Consider a path with a single hop employing the FIFO scheduling discipline. On such a path, $D(t)$ (delay of a hypothetical packet of size s bytes) is equal to the sum of queue size at time t, $Q(t)$, and the minimum delay (consisting of the propagation and transmission delay), i.e.,

$$D(t) = Q(t) + \frac{s}{C} + P \tag{3.3}$$

where P is the propagation delay and C is the hop capacity. Note that $Q(t)$ is defined in units of time, i.e., the amount of time required to service all remaining bytes in the queue. $Q(t)$ can be calculated using traces of all packet arrivals and departures along with hop capacity and propagation delay. Three properties of $Q(t)$ are crucial to this calculation.

- When the queue is draining, $Q(t)$ decreases linearly with slope -1. This is because we define $Q(t)$ in terms of *time* remaining to drain the queue.
- When a new packet arrives, $Q(t)$ increases (by the transmission time of that packet) instantaneously thereby causing a discontinuity.
- Once the queue becomes empty, $Q(t)$ stays constant at 0.

These three properties are shown in Figure 3.2. Notice that $Q(t)$ in a finite time interval is easily

described using a finite set of points where the slope changes (from/to 0, -1 and ∞). Hence, $Q(t)$ in a finite interval can also be efficiently stored in a data structure. Furthermore, it is easy to calculate statistics such as the mean, Cumulative Distribution Function (CDF) of such a piecewise linear curve. Hence, in the single-hop case, statistics of $D(t)$ are also easily calculated using just packet arrivals, queue capacity and propagation delay.

The above can also be extended to a multi-hop path. Consider a multi-hop path with n hops. Denote the queue size of hop h at time t to be $Q_h(t)$. The capacity and propagation delay are denoted by C_h and P_h. Let $D_s(t)$ be the end-to-end delay of a hypothetical packet of size s injected at time t. $D_s(t)$ is the sum of the across-hop delays seen at each of the n hops. Moreover, the queue size seen at a hop h is $Q_h()$ at a time that depends on the previous hops. Hence, we have the following extension of Equation 3.3 to multiple hops

$$\begin{aligned} D_s(t) = Q_1(t) + \frac{s}{C_1} + P_1 + \\ Q_2(t + Q_1(t) + \frac{s}{C_1} + P_1) + \frac{s}{C_2} + P_2 + \dots \\ Q_n(t + Q_1(t) + \frac{s}{C_1} + P_1 + Q_2(\dots)\dots) + \frac{s}{C_n} + P_n \end{aligned} \tag{3.4}$$

To calculate $D_s()$, we first calculate all $Q_h(t)$ using the packet arrivals at the various hops. As before, these can be represented using the finite set of points where the slope changes. These are then combined using Equation 3.4. Since $D_s(t)$ is the sum of a finite number of curves $Q_h()$, it also changes slope a finite number of times and can be represented in a compact manner. This representation can be used to calculate statistics of $D_s(t)$. If the probe packets were derived from some distribution, the corresponding ground truth was obtained by convolving various $D_s(t)$ with the packet size distribution.

3.3 Internet-based Experiments

Now, we describe two sources of real-world Internet data that we used. The first source was a passive collection of cross-traffic arrivals at *all* interfaces of a router. The second source was a novel active experiment where we injected packets into an operational network and simultaneously collected packet traces.

Figure 3.3. The active packet injection device, the IXIA 400T (left) that we used. (Courtesy: IXIA [Ixi])

3.3.1 Router Traces

To aid our investigation into estimating cross-traffic, we used data from a "full router" experiment [HVPD04]. In this experiment, all interfaces of a router in the Sprint Internet backbone were monitored using DAG [End] cards. The router had two `OC-48` links of capacity 2.4Gbps, three `OC-3` links of capacity 155Mbps and one `OC-12` link of capacity 622Mbps each. As is discussed in [HVPD04], one of the `OC-3` interfaces is highly utilized ($50 - 70\%$) and mostly carries traffic input via the `OC-48` links. For evaluating our cross-traffic estimators, we used the arrival processes to this `OC-3` link as our cross-traffic process.

3.3.2 Active Experiments

The second type of Internet-based experiments that we used consisted of active injection of packets and simultaneous capture. We chose an `OC-192` link (capacity of 10Gbps), in a large tier-1 ISP, that was reasonably utilized (around 50%, i.e., $\rho = 0.5$). We injected packets across this router through a packet injection device. Our active injection device was an Ixia 400T [Ixi] (see Figure 3.3), a specialized hardware device that is typically used to test routers. This device can be programmed to send packets of different sizes, header fields at specified times. The programming

interface that we used was a *Tcl* command-line interface that could be remotely invoked. This allowed us to generate our probe streams.

The Ixia 400T sent packets on an `OC-12` link that was connected via an un-utilized `OC-48` link to our chosen router. Packets were addressed to suitable IP addresses so that they would be output on the chosen `OC-192` link of this router. The `OC-12` input was necessary as a way to control the maximum load, in the event of misconfiguration. The cross-traffic was not controlled in any way.

We monitored the input OC-48 and output OC-192 links using GPS-synchronized DAG [End] cards. This provided us with the arrival and departure times of the probe packets. The output link monitor also provided us with the departure timestamps of the intervening cross-traffic. These timestamps were accurate to sub-microsecond levels.

3.4 Summary

In this chapter, we first described the methodology that we follow in this dissertation. Our methodology consists of the use of rigorous probability and queueing theory, simulations to fine-tune and validate theory and real Internet experiments. The rigorous theory that we described consisted of simple, but useful concepts of independence between random variables and performance metrics of estimators such as bias, variance and MSE (mean square error). We described various simulation frameworks we used. These included existing simulators such as *ns-2* [Sim] and newer simulators developed from scratch by us. We also described a novel Ground Truth Calculator (GTC) that uses traces of packet arrivals to hops of a path to calculate the underlying ground truth of the system, e.g., queueing delay distribution of a hop. Finally, we described the two different kinds of router traces that we used. One of them consisted of all packet arrivals to a router while the other captured all packets transmitted on a link. The latter was novel because we simultaneously injected probe packets.

Chapter 4

Sampling: The Role of PASTA

"The universe as we know it is a joint product of the observer and the observed."

- Teilhard de Chardin, French Jesuit, paleontologist, and philosopher

In this chapter, we investigate sampling-related issues of active network measurements, i.e., how do we choose the sending times of probe packets and why? For this chapter, we assume that the observed and desired properties are the same, e.g., end-to-end delay. Problems for which this assumption is not true typically require a complex inversion step that derives the desired property from the observed property. The results of this chapter are also relevant for them. We defer that discussion to the next two chapters. Our starting point here is the current state of art, the often-used "Poisson Arrivals See Time Averages (PASTA)" property [Wol82]. PASTA has been used to justify the sending of probe packets (pairs and trains, too) at Poisson epochs in an effort to obtain unbiased estimates of properties such as end-to-end delay [Pax99], loss [Pax99], available bandwidth [SKK03], bulk throughput [Pax99], etc. To clarify what Poisson probing, and PASTA can and cannot provide for active measurements, we start by mapping out the various issues involved in choosing a probing stream. We find that two kinds of biases, *sampling bias* and *inversion bias*, together with variance determine the optimal probing stream. Sampling bias may occur when "phase-locking" between the probes and cross-traffic causes certain system behaviors to be observed less or more frequently than they actually occur. Inversion bias occurs because the observations obtained via probing include the impact of probes as well. Note that we use the term "inversion" since removing the impact of probes

is equivalent to obtaining the desired property from our observations. Variance is caused because estimation is typically performed by observing a finite number of probe packets. However, PASTA only states the optimality, with respect to sampling bias, of individual probe packets (not pairs or trains) sent at Poisson epochs. PASTA neither offers any guidance into understanding inversion bias nor does it state anything about the estimation variance achieved with Poisson probing.

To fully understand all these issues, we leverage the theory of Palm martingale calculus [BB03], which is a modern framework to study sampling-related questions in queueing theory, especially the relation between event averages (what a probe might observe) and time averages (the underlying ground truth). We first consider systems that have no inversion bias, i.e., hypothetical measurement scenarios involving zero-sized virtual probe packets (pairs or trains). We show that such non-intrusive probe streams will also have zero sampling bias if they or the cross-traffic enjoy the so-called *mixing* [Pet83] property. The mixing property is satisfied by many non-Poisson streams including those with arbitrary i.i.d. (independent and identically distributed) random inter-arrival times. Then, we extend these insights to study systems with inversion bias, i.e., practical measurement scenarios with probe packets of non-zero size. We find that there is no known way to remove inversion bias apart from sending well-spaced probes (pairs or trains). Such a *rare probing* setting is easily viewed as an approximation of non-intrusive probing and therefore, justifies the application of the non-intrusive results. Hence, our final recommendation is that rare probes (pairs or trains) be sent at epochs that are *mixing* to avoid sampling bias. We also show, via simple examples, that non-Poisson probing streams can outperform Poisson streams. Though we only consider end-to-end delay and related functions such as jitter, our results are also applicable to loss-based metrics.

This chapter is organized as follows. In Section 4.1, we provide a basic overview of PASTA and related work in active measurements. Using this as motivation, we define the scope of our problem and provide an overview of our contributions. In Section 4.2, we use the estimation of end-to-end delay in simple one-hop systems to map out key issues, e.g., the two kinds of biases, estimation variance, and our key results, e.g., achieving unbiased sampling with mixing probing streams. In Section 4.3, we introduce powerful mathematical machinery from Palm martingale calculus [BB03] and use it to prove results on non-intrusive probing achieved with virtual (zero-sized) probes. Using simulations, we confirm that our results are valid for the tandem (First-In First-Out (FIFO)) route

models commonly used in active measurements, as well as considerably more realistic models incorporating correlation of cross-traffic across hops, and cross-traffic with feedback, e.g., TCP. In Section 4.5, we extend the results on non-intrusive probing to the practical case of non-zero sized probe packets. We derive our final recommendation of *rare probing* at mixing epochs as an effective way of minimizing both biases. In Section 4.6, we summarize our contributions, namely, the development of rare probing as a sound practical strategy given the limited role of PASTA in designing active measurements.

4.1 Overview

In this section, we first discuss prior work on the "Poisson Arrivals See Time Averages" (PASTA) principle especially in the context of network measurements. We use this to motivate our problem and define the scope of our investigation. We end this section outlining our contributions.

4.1.1 Motivation - PASTA

Poisson Arrivals See Time Averages (PASTA), is a property applicable to many stochastic systems. In essence, it states that observations made of a system at time instants obeying a Poisson process, when averaged, converge to give the "true" value, which is the average that an ideal observer would make when monitoring the system continuously over time. A classical queueing theory example is that the mean number of customers waiting in a queue as seen by Poisson arrivals is equal to the (time-averaged) mean number of customers waiting. PASTA was first formalized by probabilists, notably in the 1970's. Wolff, in his classic 1982 paper [Wol82], unified and extended the then-existing PASTA results. The generality of his formulation, based on the "Lack of Anticipation Assumption (LAA)", which requires simply that the past history of the system not influence the arrival times of future observers, did away with the need to prove ergodic theorems for each new system, and led to PASTA being widely used.

PASTA has been used to justify the sending of probe packets [Pax97a], pairs [SKK03], trains [LRLL04] at Poisson epochs in an effort to obtain unbiased estimates of quantities of interest. Pois-

son sampling has been applied for quantities as varied as connectivity [Pax97a], end-to-end delay [Pax99], loss [Pax99], available bandwidth [SKK03], bulk transfer [Pax99]. Despite the generality of the PASTA result of Wolff and its widespread use in network measurements, the role and utility of PASTA for active measurements has never been thoroughly addressed in the theoretical and practical senses (see [TDDA05] for basic empirical questions in this context, though).

4.1.2 Scope of Investigation

As discussed earlier, we use the sampling-inversion viewpoint of active measurements. Our aim, in this chapter, is to investigate fundamental questions that arise in the context of sampling. Specifically, how should we choose the sending times of probe packets? By investigating the issues that impact this choice, we aim to develop a generic recommendation to guide the designers of active measurement techniques. The issues we investigate are of relevance to all measurement problems. However, the final recommendation depends on whether the desired and observed property are the same or not. Hence, we find it convenient to assume, for this chapter, that the goal is to estimate statistics about a desired property, say, end-to-end delay, using observations of the same property obtained via probes. In the next chapter, we consider measurement problems with a desired property that is different from the observed property. The insights obtained from this chapter are also used later to explain the fundamental limits of non-intrusive active measurements.

The observed (and desired) property that we consider are end-to-end delay and multi-dimensional delay-based functionals such as jitter and packet train dispersion. Delay is a simple, yet important observable in its own right and, apart from loss, forms the basis of all active measurement inferences. The simplicity of delay allows rigorous results to be derived, and yet it provides a context rich enough to inform the rest of this dissertation and active measurement techniques in general.

The key questions that we address in this chapter are given below. PASTA is the starting points for many of these questions given the traditional emphasis on Poisson probing.

- When is PASTA valid in the strict sense?

- When and in what sense is PASTA useful when it holds? Is Poisson probing necessarily optimal?
- Are there cases when Poisson probes should *not* be used? If so, how should probes be sent?
- What role is played by PASTA within the inference problems of network measurements?
- What does PASTA apply to? In other words, Poisson Arrivals See Time Averages, but of what? Does PASTA hold for *any* quantity that may form the object of active measurement?

As is usually the case [Pax99, ZDP01], we assume that the network is *stationary*. This means that the system conditions do not change over time. For practical purposes, we require that they do not change over common measurement timescales [ZDP01]. We also assume that the network conditions and cross-traffic is *ergodic*. This implies that network conditions, during the times at which measurement is done, are representative of the average-case network behavior. Rigorous derivation of results like PASTA require this condition to be true. Finally, we assume that the probe packets are generated as an stream independent of cross-traffic. This is natural since the experimenter controls the probing stream.

4.1.3 Contributions

In this chapter, we investigate the universal question faced in designing any active measurement technique - at what times do we send the probe packets, pairs or trains? As mentioned earlier, we assume, in this chapter, that the observed and desired properties are the same. For such measurement problems, we find that general methods to remove the impact of probes from observations (an inversion step) is hard. This necessitates *rare probing* and consequently, non-intrusiveness becomes a necessity. This motivates us to first gain insights into hypothetical non-intrusive measurement scenarios involving virtual probes of zero size and later, extending these to practical measurement scenarios involving real probes of non-zero size. We give rigorous answers using Palm martingale calculus [BB03], a modern branch of queueing theory that studies precisely the questions that we are interested in - the relation between what an observer sees and the underlying “true” average. We also illustrate our results using simulation experiments.

The issues that we investigate are also relevant for measurement problems with a desired property that is different than the observed property. This is discussed in the next chapter. The issues in this chapter also help us understand, in Chapter 7, fundamental limits to active measurements that arise due to biased sampling. Our specific findings from this chapter group naturally under three distinct categories, and can be summarized as follows.

Sampling Bias versus Intrusiveness

- PASTA states that Poisson sampling is unbiased. In the non-intrusive case, we show that this is not unique to Poisson but is shared by a large class of other so-called *mixing* sampling processes.
- PASTA states that Poisson sampling remains unbiased even when observers are not virtual, but contribute to system load, and that this property is not shared by other sampling processes. We argue that it does not follow that Poisson is superior, because of variance and inversion issues, described next.
- We show that *rare probes* can be used to avoid issues of intrusiveness and inversion. In such a scenario, Poisson is no longer special and the only relevant issue is that of variance.

Bias versus Variance

- PASTA is a statement about bias. It is silent on variance, which is nonetheless of equal importance to estimation.
- There is no general result stating the optimality of Poisson observations with respect to variance or Mean Square Error (MSE), except asymptotically for MSE in the intrusive case[1]. Indeed, optimality would in general require a probing stream which is well matched in some sense to network characteristics. In Section 4.2.3, we give explicit examples showing that Poisson probing can be sub-optimal.

[1] Recall that MSE $=$ bias^2 $+$ variance.

Sampling versus Inversion

- To obtain "what one wants" from what has been observed, an additional *inversion* step is required. For example, what the delay distribution would have been if the probes were non-intrusive, based on measurements that were free of sampling bias but which were intrusive. Inversion is typically complex, and in general impacts both bias and variance.

- PASTA is silent on inversion. There is no result stating that Poisson sampling is unbiased, or otherwise optimal, for the full problem of sampling followed by inversion. Furthermore, the zero sampling bias of Poisson in the intrusive case is not necessarily an advantage when it assists in measuring the wrong quantity. It may even be that inversion is impossible, in which case Poisson sampling cannot magically provide unbiased estimates.

The picture that emerges is that PASTA plays only a very restricted role in network measurements. In a nutshell, estimation based on active network measurements seeks to optimize total bias as well as variance performance, and must therefore address both sampling and inversion issues. PASTA deals only with sampling of the available observable, not with inversion to the final quantity of interest, is ignorant of variance, and furthermore excludes the low variance potential of alternative schemes which also enjoy zero sampling bias. In contrast, it's strength, a lack of sampling bias even in the intrusive case, is not necessarily relevant given the near universal need for inversion. Simply stated, PASTA is useful only when the goal is to send *individual* probe packets *and* there is no need to remove the impact of these probes from the observed delays.

4.2 PASTA and Delay: the Issues

In this section, we illustrate the key facts and issues involved in measuring end-to-end delay using probe packets, and the role of PASTA, in the simple context of a single, FIFO queue fed by probe traffic and cross-traffic obeying simple models. We start by describing this experimental setup in Section 4.2.1. This includes the cross-traffic and probe arrival processes that we used. In Section 4.2.2, we illustrate how and when two different kinds of biases, sampling bias and inversion bias, arise. Using hypothetical virtual probes of zero size and realistic probes of non-zero size, we

show that the zero sampling bias is exclusive to Poisson probing only because of the intrusiveness of probing. We also show the flip side, that is, intrusiveness also causes inversion bias that is non-zero for all probing streams including Poisson streams. In Section 4.2.3, we use illustrative examples to show that the variance of Poisson probing may be so large that non-Poisson probing streams with non-zero sampling bias yield better performance. Finally, in Section 4.2.4, we illustrate one of our key results - that zero sampling bias in the virtual probe case requires the probing stream or cross-traffic to possess the so-called *mixing* property.

4.2.1 Experimental Setup

To conduct the single-hop experiments presented in this section, we used our queueing simulator, *qsim*. Recall that *qsim* implemented the Lindley recursion for a FIFO queue (for example, see [BB03]) and was implemented in `C` and `Matlab`. We used a queue of capacity 10Mbps and zero propagation delay. Two kinds of statistics were collected. First, per-packet delay values, from which the delays of probes and cross-traffic were extracted. Second, our *Ground Truth Calculator (GTC)* was used to obtain the ground truth distribution D of the delay of probe-sized packets in the system. As it was stored in histogram form, this created a discretization error. However, the size of the histogram bins was chosen to be small enough that these errors were negligible on the scale of the plots given. Similarly, we used long simulations of 1000000 probes to make confidence intervals small or negligible (in the latter case we don't show them), and employed large warm-up periods to damp transients.

For the single-hop (with no propagation delay), the ground truth of delay, D, is obtained by convolving the probe size distribution with the queue size (or waiting time) distribution, W. W may also be viewed as the *virtual delay process* of queueing theory in such a one-hop case. Virtual delay is the stochastic process $W(t)$, defined for continuous time $t \in \mathcal{R}$, which corresponds to the waiting time a packet of size $x = 0$ would experience when sent at time t into the network in steady state. Because this is also the delay for such a hypothetical zero-sized packet, we refer to this as the *virtual delay*. This process represents the ground truth of the delay (of zero-sized packets) in the system.

M/M/1 Queue

As several of the examples employ the M/M/1 queue, we summarize some relevant properties here. In the M/M/1 system, packets arrive as a Poisson process of rate λ, and each takes an exponential amount of time, with parameter (and average) μ, to be transmitted. To ensure stability and stationarity of the system, we require the system utilization $\rho = \lambda\mu$ to satisfy $\rho < 1$. It turns out [Kle75b] that the time a packet spends in the system, i.e., its end-to-end delay, is also exponentially distributed with parameter $\bar{d} = \mu/(1-\rho)$:

$$F_D(d) = \mathrm{P}(D \leq d) = 1 - e^{-d/\bar{d}}, \quad d \geq 0, \rho < 1 \tag{4.1}$$

with mean $E[D] = \bar{d}$, where D is the random variable representing the delay of a given packet.

A related but distinct quantity is the waiting time W of a packet, which is also the *virtual delay experienced by a zero-sized packet*. This distribution:

$$F_W(y) = \mathrm{P}(W \leq y) = 1 - \rho e^{-y/\bar{d}}, \quad y \geq 0, \rho < 1, \tag{4.2}$$

with mean $E[W] = \rho\bar{d}$, has an atom at the origin corresponding to the probability $1-\rho$ of finding the system empty, resulting in zero waiting time (and zero delay in the case of a zero-sized packet).

Probing Streams

We used the following 5 arrival processes as probing processes. These represent a spectrum of "bursty" behavior. In each case, we also list the distribution of inter-arrival times assuming that the average is μ:

- **Poisson:** Exponentially distributed inter-arrival times τ yielding a Poisson process. This is a *renewal process*, i.e., the inter-arrivals are independent and identically distributed (i.i.d) random variables.

$$P(\tau \leq T) = 1 - e^{-T/\mu} \quad 0 \leq \tau.$$

- **Uniform:** Inter-arrivals distributed uniformly in an interval $[\mu - a, \mu + a]$ ($[0, 2\mu]$ by default). This is also a renewal process.

$$P(\tau \leq T) = \frac{1}{2} + \frac{T-\mu}{2a} \quad \mu - a \leq \tau \leq \mu + a.$$

- **Pareto:** Heavy tailed Pareto distribution with finite mean and infinite variance. This is also a renewal process.

$$P(\tau \leq T) = 1 - \left(\frac{\tau}{(D-1)\mu} + 1 \right)^{-D} \quad 0 \leq \tau.$$

We use $D = 1.5$.

- **Periodic:** Constant inter-arrival times. This is also renewal but in a very degenerate sense as inter-arrivals are constant, and is best regarded as a deterministic stream. The random phase, determining the offset of the periodic grid from the time origin, makes it stationary despite its rigidity.

$$P(\tau = T) = 1.$$

- **Exponential (first order) Auto-Regressive 1 (EAR(1)):** The EAR(1) [GL80] is a point process that, like the Poisson process, consists of exponential inter-arrival times of intensity $\lambda = 1/\mu$. But, unlike the Poisson process, inter-arrival times are correlated because consecutive inter-arrivals (τ_n and τ_{n-1}) are related to each other via an i.i.d. sequence of exponential random variables (E_n) of mean μ and i.i.d. sequence of Bernoulli random variables (I_n) of mean $1 - \alpha$,

$$\tau_n = \alpha\tau_{n-1} + I_n E_n, \quad \alpha \in [0, 1).$$

The correlation structure is that of a positively correlated AR(1) process, namely

$$\text{Cov}\,(\tau_i, \tau_{i+j}) = \alpha^j, \quad j = 0, 1, 2, \ldots. \tag{4.3}$$

The time constant of this geometric decay translates to a correlation time scale of $\tau^*(\alpha) = (\lambda \ln(1/\alpha))^{-1}$, which rises from 0 when $\alpha = 0$ (the Poisson case), to ∞ as $\alpha \to 1$. The EAR(1) process, as described above, is not i.i.d. because it has correlated inter-arrivals with exponential marginal.

4.2.2 Bias

In this section, we will use simple examples to illustrate sampling bias and inversion bias. To show that zero sampling bias is exclusive to Poisson streams only because of intrusiveness, we first consider hypothetical zero-sized probes. We find that all of our probing streams have zero sampling

bias in this case. Then, we illustrate PASTA, i.e., this bias remains zero only for Poisson probes when the probe packets are made intrusive. Finally, we illustrate the flip side of intrusiveness - it causes an inversion bias that is not zero for Poisson streams.

Sampling Bias in the Non-intrusive Case

Figure 4.1 gives results for each of the above probing streams, with a shared average inter-probe spacing of 10ms, using probes of zero size. Though zero-sized probes are not practical, they help us understand the issue of sampling bias in isolation. This is because there is no issue of intrusiveness (probes do not affect the system) or of inversion (we are directly measuring what we wish to measure). We use 500 bytes as the average size of cross-traffic packets and 0.4 as the utilization of the single hop.

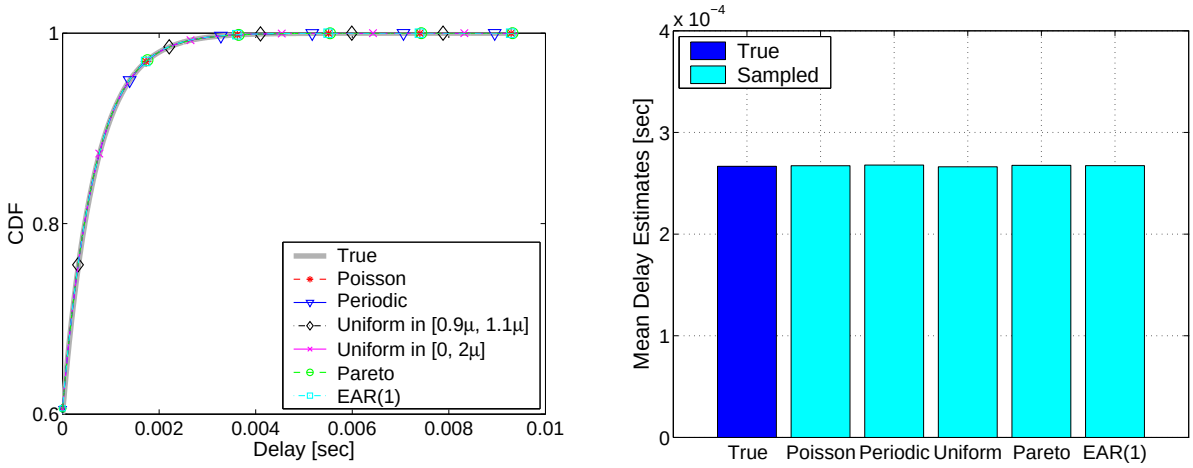

Figure 4.1. Sampling bias of delay, non-intrusive case (probe size $x = 0$). Left: CDF as seen by various probing streams, and the true delay distribution. Right: resulting mean estimates. We do not show the confidence intervals since they were very small. Each probing stream is unbiased.

The grey curve in the plot on the left of Figure 4.1 shows the true Cumulative Distribution Function (CDF) for the delay of zero-sized probes, calculated from Equation 4.2. As expected, the curve corresponding to Poisson probes agrees with the true one: as is well known, PASTA applies to this system. However, the other five curves overlay the true result equally closely. In this non-intrusive case, the lack of sampling bias of Poisson probing *is shared by many other probing schemes*. The (tightly estimated) expected delays in the right plot of the figure confirm this by agreeing with the true value in each case. In Section 4.3, our main result is to prove that a wide

class of processes share this desirable property, and for far more general systems (including most systems of practical interest) than the simple M/M/1 queue.

Sampling Bias in the Intrusive Case

In Figure 4.2 we consider the same probing arrival streams, but allow the probes to have a transmission time $x > 0$ (x is taken to be a constant 500 bytes for simplicity, but this is not essential). As a result, intrusiveness becomes an issue: probes do affect the system, both in load and in more detailed characteristics. To avoid dealing with inversion issues, we assume that our objective is to know the true delay, in the full system combining cross-traffic and non-virtual probes, that a packet of transmission time x would experience. In other words, we still seek to measure the same object that we have direct access to, through probing.

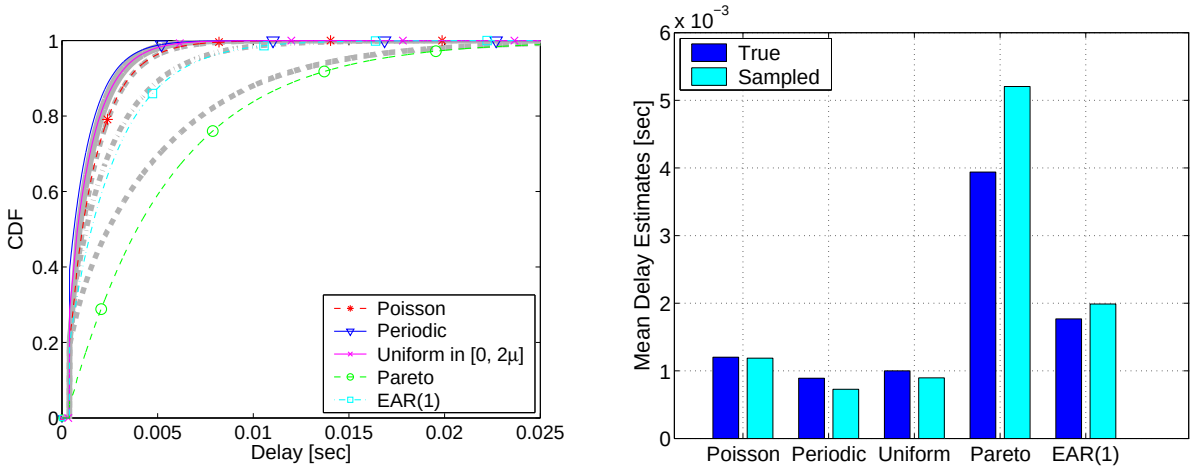

Figure 4.2. Sampling bias of delay, intrusive case ($x > 0$). We do not show results for the "Uniform in $[0.9\mu, 1.1\mu]$" probing stream, to avoid clutter, since they mirror the other results. Left: CDF as seen by various probing streams, and true delay distributions (one per stream, the closest grey curve in each case). Right: resulting probe based mean estimates, and true means. Again, we do not show confidence intervals since they were very small. Each probing stream results in a new true delay distribution, which is sampled with bias by the probes, except the Poisson case (PASTA).

The mean estimates in the right hand plot of Figure 4.2 confirm that each probing stream results in a different system behavior (despite equal loads), and shows that each now gives a biased estimate of its respective $E[D]$, with the exception of "Poisson". Hence, PASTA continues to hold, whereas the other probing streams, despite being unbiased when $x = 0$, now suffer from a bias due to intrusiveness. The corresponding CDFs in the left hand plot show in greater detail how the systems

are different for each stream, and how the bias varies as a function of delay. These results illustrate that PASTA holds in the intrusive case. In Section 4.5, we will state the general conditions under which PASTA can be expected to hold.

Inversion Bias

We now study inversion bias in isolation. We achieve this by employing Poisson streams exclusively, thereby benefiting from their zero sampling bias in all cases. Furthermore, we let the probe transmission time x be exponentially distributed with the same mean, μ_T ($= 500$ bytes), as for the cross-traffic packets. This results in a combined system which is still M/M/1, with rate $\lambda = \lambda_T + \lambda_P$ and average transmission time μ_T, enabling Equation 4.1 to be used. We use probing traffic of different levels of intrusiveness which increases the link utilization in increments of 0.1 from 0.4 to 0.8.

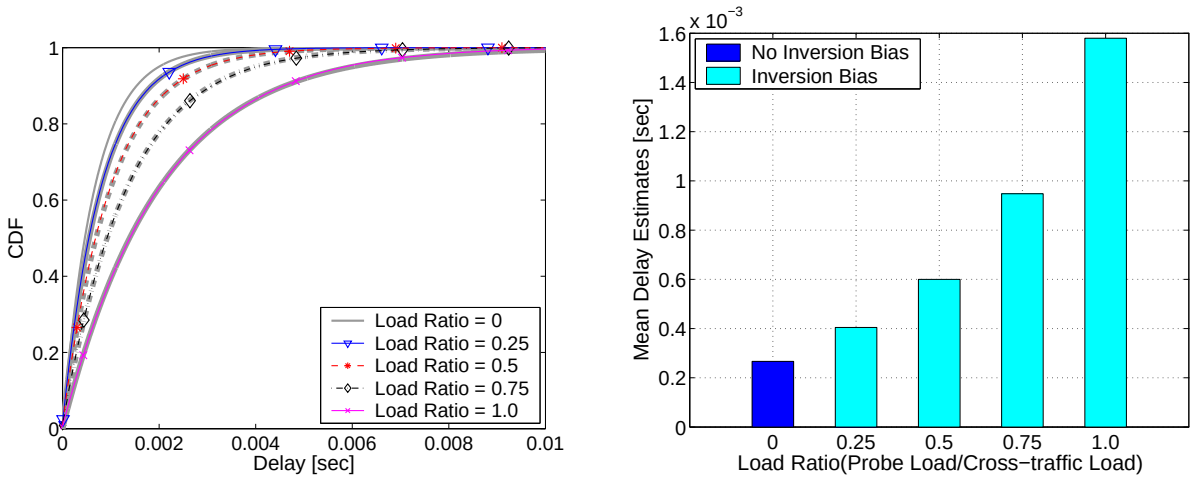

Figure 4.3. Inversion bias of delay, range of intrusiveness ($x \geq 0$). Left: CDF as seen by Poisson probing streams of different rate, and true delay distributions (one per stream), as well as the true un-perturbed delay (with no probes). Right: corresponding mean delays as a function of probe to total load ratio. PASTA eliminates sampling bias, but total system behavior increasingly deviates from that of the un-perturbed system.

Figure 4.3 shows the unsurprising but significant fact that increasing the probing load through increasing λ_P results in the overall system deviating increasingly far from the original unperturbed system in which $\lambda_P = 0$. Consequently, even if an estimate of the true mean (or the CDF) is unbiased, that estimate is an estimate of the **full** (probe + cross-traffic) system, **not** the unperturbed

(cross-traffic only) system that one wishes to measure. Thus, what we want is not what we directly measure. To obtain the desired unperturbed delay from the observations of perturbed one requires an entirely separate inversion step, which, even in this very simple example of inverting one kind of delay to another, is highly non-trivial. For other inference objectives common in active measurements, such as using packet-pair methods to estimate available bandwidth, the degree of inversion required, and therefore its potential impact, is far greater. Another way of seeing this is to note that probes sent as a Poisson process at the sender will not arrive as Poisson process at the bottleneck link in general, and will also be affected by their onward passage from that link to the receiver. Thus, the probes are sampling the bottleneck link, but not in a Poisson way, and not in isolation.

We have two contributions to make on the inversion issue. First, we point out that not only can it be extremely challenging, but more importantly that it is an inherent difficulty for which PASTA offers no solution. To our knowledge, there are no known general inversion techniques to remove the impact of probe packets. In the next two chapters, we consider inversion of a different kind, using delay observations to estimate properties of cross-traffic. We find that, even in a simple one-hop system, unless the cross-traffic obeys certain assumptions, full knowledge of the cross-traffic process feeding the hop is unobservable. In such a case, strict inversion is impossible even in principle. This serves to illustrate the hardness of any kind of inversion which PASTA is powerless to mitigate. Second, in spite of the difficulties described above, there is a general way, the *rare probing* strategy to avoid inversion bias in the intrusive case, which we explore in Section 4.5.

There is a substantial literature on *perturbation analysis* (see [Gla91]) which addresses the problem of determining the behavior of a perturbed system from that of the unperturbed one. However, there are no immediate or simple answers to the difficult inversion problems of active measurements.

4.2.3 Bias versus Variance

So far, we focused on the bias of estimators based on a simple average of delays experienced by probes. We now look into the variance of these estimators. When bias is non-zero, we examine bias-variance trade-offs in the traditional manner by considering the mean squared error (MSE) $\text{MSE} = \text{bias}^2 + \text{variance}$. To illustrate only the effect of estimation variance, we ignore inversion

bias. In other words, the probe-derived estimates are compared to the ground truth in the perturbed system that includes the probes.

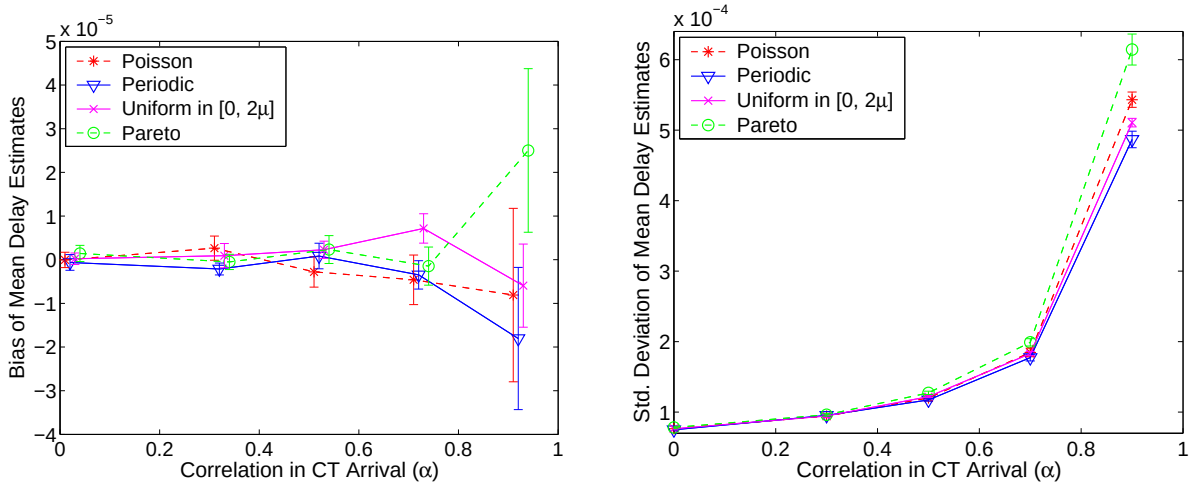

Figure 4.4. Bias and variance of delay with correlated cross-traffic, non-intrusive case ($x = 0$). Left: bias of mean estimates seen by different probing streams as a function of the EAR(1) parameter α of the cross-traffic, using 100000 probes. Right: corresponding estimates of standard deviation. Although all probing schemes are unbiased, their variances differ, and Poisson is **not** the smallest.

Thus far, we have considered cross-traffic packets arriving as a Poisson process. However, in general cross-traffic will interact with probe traffic in ways which depend on the correlation or "burstiness" structure of each, and estimation variance will be a function of these interactions. To show this clearly we need a richer context than the simple "memoryless" structure of the Poisson process. We use the EAR(1) [GL80] process introduced earlier as a convenient way to generate a point process which has a well defined correlation time scale.

Figure 4.4 shows the effect of increasing α (and hence, short-time correlation) on the estimation of mean delay, for four different non-intrusive probe streams of identical rate. In the left plot we see the expected lack of bias for each stream (note the vertical scale and confidence intervals, offset horizontally for visibility), in agreement with the results of Figure 4.1, regardless of the value of α. In contrast, the right hand plot shows that the standard deviation of the estimates separate at large α. This separation clearly exceeds the confidence intervals: the Poisson stream has higher variance then either Periodic or Uniform. This is a counter-example making the important general point that Poisson sampling does not imply minimal variance.

We now offer some insight into why Poisson probing gives rise to higher variance than Periodic

in this case. First note that as the correlations in the cross-traffic increase, so do those of the virtual delay process $W(t)$ itself. If we could make estimates based on $W(t)$, they would therefore have increased variance[2]. Each probing scheme samples $W(t)$, experiences the larger correlation at higher α, and thereby inherits the larger variance. Exactly how much variance however depends strongly on the details of the sampling scheme, not merely on the average sampling rate λ_P. Bearing in mind that samples which are closer together will be more correlated, periodic probing has the advantage of guaranteeing a minimum distance between them. It can therefore "jump over" correlation-inducing bursts, provided that $1/\lambda_P$ is large compared with the correlation scale of $W(t)$. In contrast, in a Poisson process arrivals may be much closer than $1/\lambda_P$ with appreciable probability, increasing the correlation considerably between such samples. In the example here, $1/\lambda_P \approx 20\tau^*$ even for $\alpha = 0.9$, so the periodic stream produces close to i.i.d. samples in all cases.

In Figure 4.5 we consider the intrusive case for a wider range of candidate probing schemes. We fix α at 0.8, and examine dependence as intrusiveness is increased by increasing probe size, shown as a function of the fraction of probing load to total load. The leftmost plot shows that (sampling) bias is now present, and increases with α, for all schemes except for Poisson (by PASTA). The variance results of the middle plot echo those seen in Figure 4.4: there are schemes which perform both better and worse than Poisson. The rightmost plot in Figure 4.5 combines bias and variance, and we see the trade-offs at work: the relative overall performance of different schemes changes with α. In particular, as bias becomes stronger for its competitors at, load ratios above 0.12, Poisson begins to outperform Periodic. However, it continues to be outdone by the Uniform renewal with wide support, even though we ignored inversion bias.

In this section, we have presented only a few illustrations of what is a general point: PASTA is silent on estimation variance, and the performance of Poisson probing, for general cross-traffic processes, plays no privileged role with regard to variance. The same holds true for MSE, with one exception. Asymptotically, as the number of samples tends to infinity the variance of any consistent estimator will tend to zero, resulting in the asymptotic MSE being equal to the bias squared. In the intrusive case, this clearly gives the advantage to Poisson probing. In general however, overall statistical performance is a function of how well the probing stream is adapted to the cross-traffic,

[2]It is well known [Cox84] that the variance of the sample mean calculated over a time window of given width is essentially the integral of the correlation function over the corresponding range of lags.

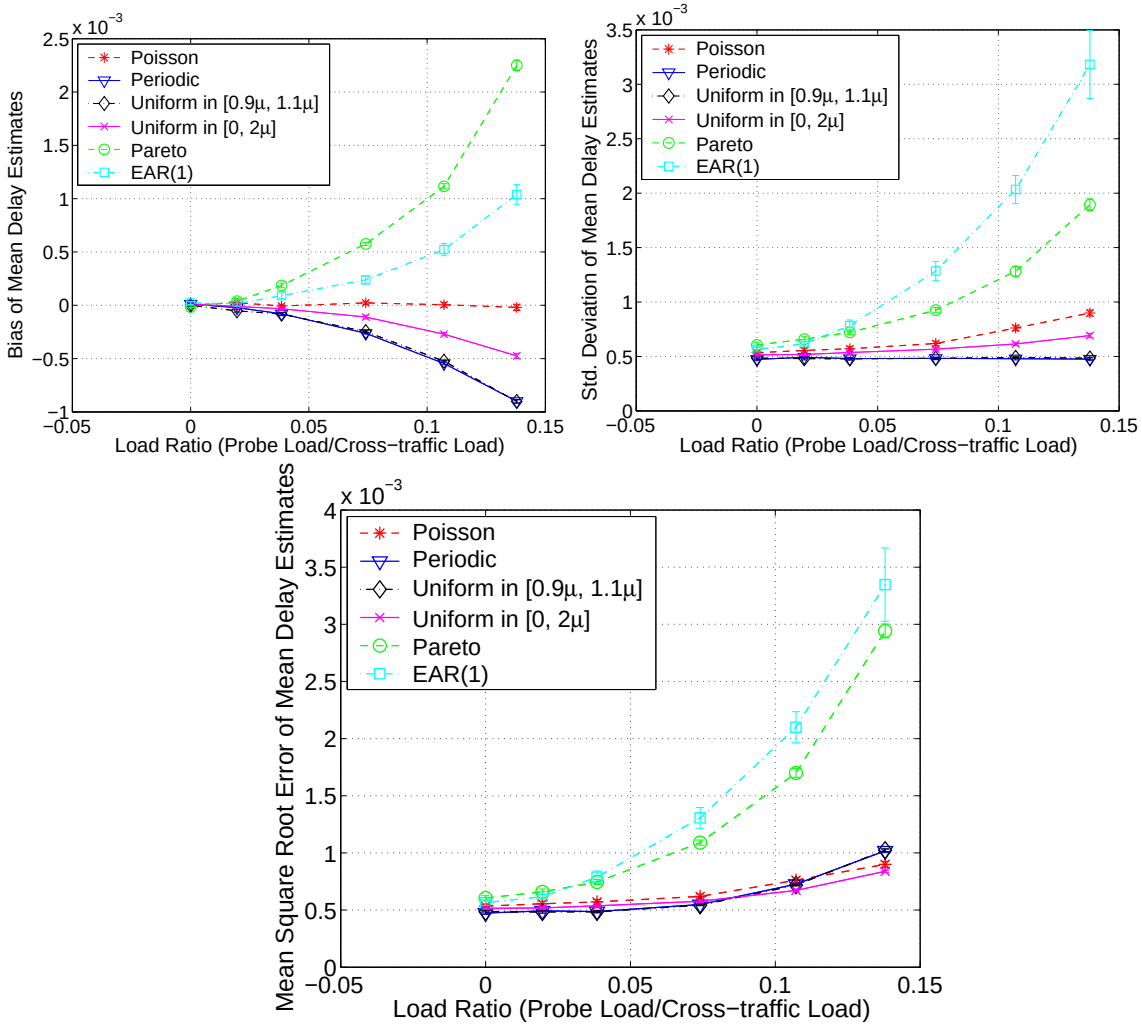

Figure 4.5. Bias, Standard Deviation and (Root) MSE of delay with correlated cross-traffic, intrusive case ($x > 0$). Left Top: Bias of mean estimates seen by different probing streams for $\alpha = 0.8$ as a function of intrusiveness probe load/total load. Right Top: Corresponding estimates of standard deviation. Bottom: Corresponding $\sqrt{\text{MSE}} = \sqrt{(\text{bias}^2 + \text{variance})}$. Only the Poisson probing is unbiased, but the scheme with minimal (Root) MSE depends on α.

and the nature of that traffic. The optimal approach (if any) will also be strongly determined by the choice of constraints such as measurement duration, probe budget, and acceptable intrusiveness profile.

4.2.4 The Need for Technical Assumptions

Thus far, we have passed over the issue of technical assumptions. While the numerical examples offered no contradictions, we have simply assumed that PASTA holds universally. Similarly, whilst

Figure 4.2 indicated that there exists non-Poisson sampling schemes with zero bias, no comment was made on which schemes enjoyed this property.

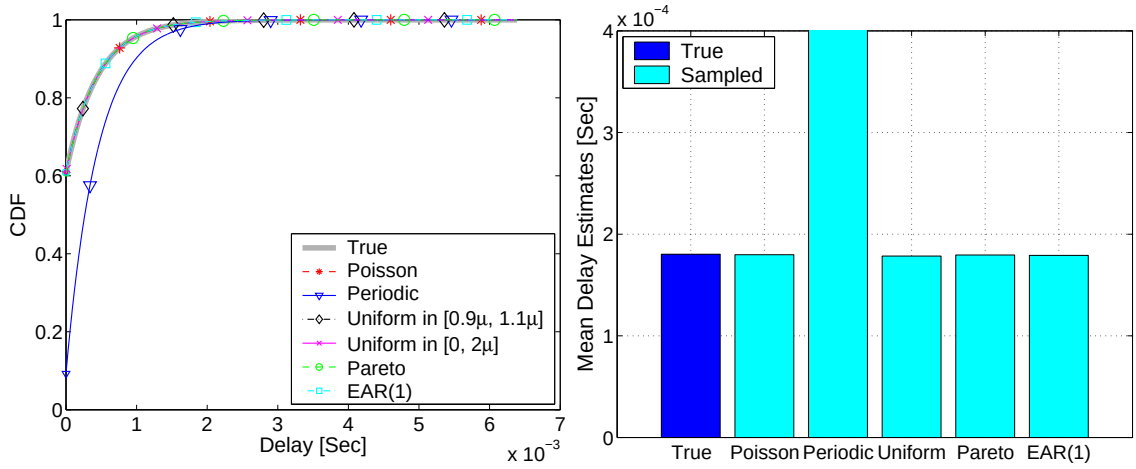

Figure 4.6. Sampling bias of delay with non-mixing cross-traffic, non-intrusive case ($x = 0$). Left: CDF as seen by various probing streams, and the true delay distribution. Right: resulting mean estimates. Each probing stream is unbiased, expect for periodic.

In fact, such results are only true given suitable *ergodicity* conditions on the cross traffic and probing processes. Even PASTA requires the system to be ergodic, i.e., the system behavior is captured during the time we measure it. It is also important to note that zero bias (across all possible measurement experiments) is in fact not our objective! To be useful in practice, we also need measurements to converge to unbiased values on a *single* sample path, as we witnessed in each example thus far. To ensure this, again suitable ergodic conditions are required. We defer a rigorous description of these issue until the next section, and complete this one with an illustrative example and intuitive explanation.

Figure 4.6 gives the outcome of a non-intrusive experiment which is identical to that reported in Figure 4.1, but with one crucial difference: the Poisson arrival times of cross-traffic have been replaced by periodic arrivals of the same average intensity (packet sizes have not been altered). We see that each probing stream continues to measure the mean delay, and even the entire delay distribution, without bias, with the exception of the periodic probe stream, which is markedly different. In fact since the period of the Periodic stream is equal to an integer multiple of the cross-traffic period (equal to 10 in this case), the two streams are effectively "phase locked", and in such a case the joint ergodicity conditions are not satisfied. As a result, the probes can never sample average

conditions on this sample path alone, but only those found at a particular point in the cycle of the cross-traffic arrivals. However despite this, since the two periodic streams are independent and stationary, the phase offset between them is random, and so estimates from the periodic probing stream are nonetheless unbiased! This is readily seen empirically if averages over *many* independent experiments are taken.

Despite the rigidity of the periodic cross-traffic, the other probing streams do satisfy the required joint conditions since they are in themselves all *mixing* processes. This is a stronger form of ergodicity. Similarly, the joint ergodicity assumptions were satisfied in the cases shown earlier of periodic probe traffic and either Poisson or EAR(1) cross-traffic, since these latter processes were mixing. In a loose sense, they provided enough variability to overcome the rigidity of the periodic probes. We discuss this in more detail in the next section.

4.3 Non-Intrusive Measurement

In the previous section we discussed sampling bias, inversion bias, variance, and contrasted the intrusive and non-intrusive cases in parallel. We saw illustrative results indicating zero sampling bias for non-Poisson streams in the non-intrusive case. We also saw that it is often desirable to make probing as non-intrusive as possible as a means to minimize inversion bias. This motivates us to fully understand the role of intrusiveness. To this end, we focus on the hypothetical non-intrusive case in this section. We leave the intrusive case and most of our comments on inversion to Section 4.5. Our aim is to expand in a rigorous way on the observations of Figure 4.1, that many processes other than Poisson enjoy zero sampling bias in the non-intrusive case. The basis of our treatment is the machinery of Palm calculus [BB03]. This is ideal for our purpose because the goal of Palm calculus is to study sampling-related issues in queueing systems. In particular, it is appropriate to study the relation between event averages (what a probe packet sees) and time averages (the underlying ground truth). Additional mathematical concepts used in Palm calculus are ergodic theory [Pet83] and marked point processes [DVJ88, BB03]. We give an overview of these areas before proving our main results. We then discuss the impact on the choice of probing streams in practice.

4.3.1 Setting

We adopt a setting which allows for very general probe traffic, cross traffic, and network models. There are three assumptions which carry over from the previous section however: *stationarity* of the probe traffic, cross-traffic and the network behavior, *ergodicity* of the cross-traffic, and *independence* of the probe traffic from the cross-traffic. Stationarity implies that network conditions do not change over time whereas ergodicity implies that the network exhibits, during the times of measurement, represents average-case behavior irrespective of the initial conditions. The assumption of independence is natural since the experimenter controls the probing stream. Later, we see that this is also necessary to ensure zero sampling bias.

We model probe traffic as a *stationary point process* $\mathcal{P}$ of intensity λ_P. That is, a sample path of the process is simply the set of times $\{T_n\}$ at which the (zero-sized) probes arrive, and there exists a probability law $P_{\mathcal{P}}$ that determines the probability of any event concerning sample paths. For example, it governs the probability that the first inter-arrival time after the time origin: $T_2 - T_1$, exceeds the mean value $1/\lambda_P$ (this would be 1 for periodic probes or e^{-1} for a Poisson stream), as well as the probability that n probes fall in a time interval I_1 and m in I_2, for arbitrary intervals I_1 and I_2. No constraints are placed on $P_{\mathcal{P}}$, we allow any structure of probe arrivals provided arrivals do not coincide.

We model cross-traffic as a *marked stationary point process* $\mathcal{T}$ of intensity λ_T. As before, this consists of the arrival times of packets, but now also *marks*, random variables associated with each packet which give additional information about the traffic. This includes first of all the random packet size, but also anything else that characterizes the stream. For example, the packet sizes may depend on the arrival patterns, or packet sizes, of previous arrivals. The probability law $P_{\mathcal{T}}$ governs all details of $\mathcal{T}$, both of arrival times and marks.

The model of an end-to-end path typically used in active probing is essentially the *tandem queueing network* of queueing theory. It consists of a set of FIFO queues and transmission links in series, each with its own independent cross-traffic stream. Packets from a given stream are all *n-hop-persistent* (traversing n hops before exiting) and frequently n is simply taken to be 1 for each stream. Our network setting does not explicitly define queues in this manner. Instead, it operates in

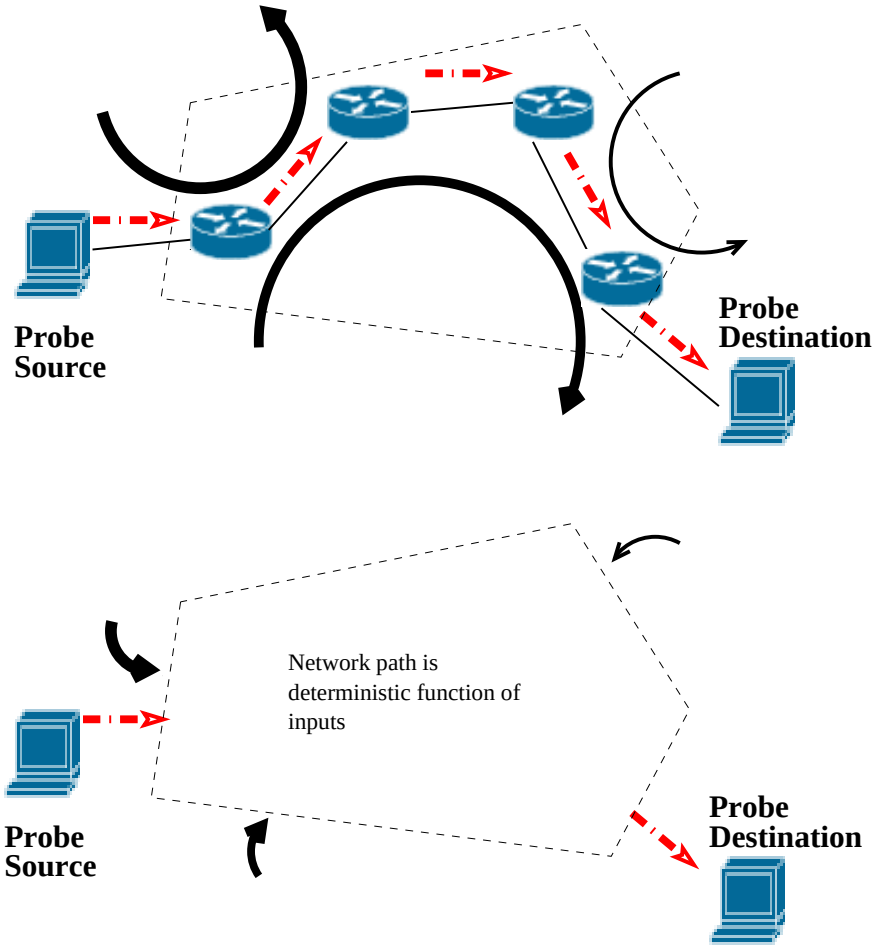

Figure 4.7. We abstract a network path into the inputs and the actual network mechanisms. The inputs are the cross-traffic process. We only specify that the actual details such as scheduling at the various hops be a deterministic function of the inputs.

a more abstract setting only requiring that everything not in $\mathcal{T}$ act **deterministically** on the cross-traffic and probe inputs (see Figure 4.7. Hence, the results that we use [BB03] and derive cover more general network settings including the following, provided the technical assumptions above are satisfied.

- Cross-traffic streams correlated across nodes.
- Cross-traffic with feedback such as TCP.
- Scheduling disciplines such as FIFO, Weighted Fair Queueing [DKS89] and Processor Sharing [PG93] which may differ from hop to hop.
- Probes which follow different paths through a network (modeling load balancing).

Technically, each of these cross-traffic streams, and their dependencies, are contained in the single marked point process $\mathcal{T}$, where the marks carry most of the detailed information, such as which nodes are traversed by a given packet. In this way, much of our general network model, in fact all of its stochastic components, is subsumed into a rich cross-traffic description. The details of the the queueing itself is not contained in $\mathcal{T}$ but would have to be specified separately if one wished to simulate the network. Our results hold true provided everything that is not in $\mathcal{T}$ acts deterministically on the cross-traffic and probe inputs.

The final component of the basic setting is to specify the *observed* property, that is the quantity related to probes that we have access to. In the case of active measurements, the available data is simply the arrival times of probes, or equivalently (since the sending times T_n are known), their end-to-end delays. Since, in this section, we consider only the non-intrusive case, the underlying observable denoted by $Z(t)$, $t \in \mathcal{R}$ can be taken to be virtual delay process $W(t)$. Recall that the virtual delay process is the delay experienced by a zero-sized packet entering at time t.

Our main goal is to learn about the process $Z(t)$. Technically, this reduces to determining the expectation $E[f(Z(t))]$ of some positive function f of $Z(t)$. The choice of f gives us great freedom in the kind of statistic we may wish to measure. Good examples already presented in Section 4.2 are the identity function giving us the mean delay or an indicator function (noting whether $Z(t)$ is smaller than some threshold) giving us the entire CDF of delay. Functions to calculate statistics such as jitter will be considered later.

With the setting established, we now indicate where sampling and inversion fit in to it. Probes sent at times $\{T_i\}$ literally sample $Z(t)$ at those times. Hence, the values $f(Z(T_1))$, $f(Z(T_2))$, ..., are what is available to estimate $E[f(Z(t))]$. As we send more probes, we have more samples and expect our estimates to improve. Specifically, we want the following *asymptotic convergence* to be (almost surely, i.e., with probability 1) true:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n)) = E[f(Z(0))]. \tag{4.4}$$

For instance, if f is the identity function, the right hand side is the mean virtual delay to which the sample mean estimate on the left hand side must converge. Stationarity implies that $E[f(Z(0))] = E[f(Z(t))]$ for any time t.

In the present non-intrusive case our aim of determining statistics of delay is not hampered by inversion issues. Such inversion is required if any quantity *other* than delay, such as available bandwidth, is the desired property. Such inversion is the focus of our next 2 chapters and uses appropriate f f to study other quantities which are related to $Z(t)$. In such cases, inversion can be seen as the task of finding the (potentially very complex) function f which can efficiently extract the desired parameter from the available delay data.

4.3.2 Ergodic Theory and Palm Calculus

Statements like Equation 4.4, where a "time mean" (the left hand side) is equivalent to an "ensemble mean" or mathematical expectation (the right hand side), are known collectively as ergodic theorems [Pet83]. Intuitively, they correspond to systems which are in some sense free enough to explore, in an unbiased way and on a single sample path, the full range of behavior which one would find if one could examine *all* sample paths. Put another way, a single sample path of an ergodic process will over time come to resemble every other sample path, with more extreme paths taking appropriately longer to emerge.

In Section 4.3.3 we determine when Equation 4.4 holds. Specifically, we show that Equation 4.4 holds (with zero-sized probes) when either the probe stream of cross-traffic satisfies the so-called mixing property. Before deriving this important result, we must first introduce key concepts of ergodic theory and Palm calculus. These formalize the probing problem and allow us to state our results formally.

The Joint Law and the Product Space

To deal with ergodicity of the system as a whole, we must know the joint law governing both probe and cross-traffic. Because these are independent, the events in the combined system can be described through the *product space* of $\mathcal{P}$ and $\mathcal{T}$, denoted by $\mathcal{F}$. It has an associated probability law P which is the product of $P_{\mathcal{P}}$ and $P_{\mathcal{T}}$.

Intuitively, $(\mathcal{F}, P)$ enumerates all sample paths of the combined system and their associated probabilities. The following example illustrates this. Consider a system in which probe and cross-

traffic are each periodic with a period of 1 time unit (for simplicity we ignore the marks of $\mathcal{T}$). Each probing sample path is completely determined by its *phase* $x \in [0,1) = T_1$, the distance from the time origin to the first probe. Similarly, the cross-traffic is described by a phase $y \in [0,1)$. Thus, each sample path of the combined system is uniquely described by (x,y), which is an element of the product space $\mathcal{F} = [0,1) \times [0,1)$. Assuming independence between the streams, the joint probability P is given by

$$\begin{aligned} P((x,y) \in [a,b] \times [c,d]) &= P_{\mathcal{P}}(x \in [a,b]) \cdot P_{\mathcal{T}}(y \in [c,d]) \\ &= (b-a)(d-c), \end{aligned}$$

where $b \geq a$, $d \geq c$, since the stationarity of T and P implies that their phases are each uniformly distributed over $(0,1]$.

Time Shifts: the θ_t Framework

For Equation 4.4 to hold, we need an appropriate form of ergodicity. One of the possible pitfalls is that it is not enough that one or the other, or even both, of the probe process and cross-traffic process be ergodic in their own right. In fact they must possess *joint ergodicity*, defined on the product space.

We begin by describing ergodicity for a single point process by means of a θ_t or *shift* operator. The shift operator represents a shift in time of value $t \in \mathcal{R}$ of the whole sample path (or set of paths) under consideration. Given such a shift, we can define the important notion of an *invariant event*.

Invariant Event: An *invariant event* of some point process is an event A such that $A = \theta_{-t}(A)$ for all $t \in \mathcal{R}$. An example of such a set is the collection of paths which have an infinite number of inter-arrivals larger than some value x, because translation would not change this property for any path in A, so $\theta_{-t}(A)$ would contain exactly the same paths.

It follows from Birkhoff's pointwise ergodic theorem [Pet83] that a point process is ergodic *if and only if all its invariant events are of probability either 0 or 1*. The statement we wish to prove, Equation 4.4, is similar to the Birkhoff theorem but in a joint setting. One can define a *product shift* that operates, simultaneously but independently, on both $\mathcal{P}$ and $\mathcal{T}$. In terms of this, our aim can be

reformulated as:

Equation 4.4 holds if and only if all sets of sample paths invariant under the product shift are of probability either 0 or 1.

To see the significance of this, we continue the "periodic-periodic" example from Section 4.3.2. Let the event A be those sample paths where, for all n, $T_n - C_n < 0.25$, where T_n and C_n are respectively the arrival time of the n-th probe and n-th cross-traffic packet. For our phase locked example, this translates to P$(x - y < 0.25) = 0.25$. However since the offset between the two streams is fixed at $x - y$ for all n, A is an invariant event, yet it has probability which is neither 0 nor 1. Hence, this system is not jointly ergodic, and Equation 4.4 will not hold.

We see that to achieve convergence of the sample based estimates, the probe sampling must not become phase locked to the cross-traffic. For any given any set of initial conditions, the combined system must be able to escape any such 'synchronization'. Conceptually, this is very similar to our intuitive understanding of ergodicity in the case of a single process. Note that such periodic behavior may actually occur in IP networks, for example when dealing with a small number of persistent TCP flows on an access link.

Palm Probability

The right hand side of Equation 4.4 can be written as

$$E[f(Z(\tau))] = \int f(Z(\tau))P(dZ),$$

which emphasizes that it is really the full law of Z as determined by P, represented by $P(dZ)$, that we would like to determine via measurement. Knowing the probability of any event A under Z is equivalent to knowing the law of Z. Let A be an event defined by some condition on $Z(\tau)$. One could choose for example $Z(\tau) \geq 1$ for some τ, but the important point is that the condition can be arbitrary. Since Z is stationary, the probability $P(A)$ of A does not dependent on τ (since Z is only a function of the cross-traffic, the probability of this event is well-defined on the product space). In contrast, the *Palm probability* $\mathcal{P}_0$ is the probability that a probe may see an event A.

$$\mathcal{P}_0(A) = \frac{1}{\lambda(b-a)} E\Big[\sum_{T_n \in (a,b]} 1_{Z(T_n)\text{satisfies}A}\Big].$$

for any positive real a, b. Recall that λ is the average rate of the (non-intrusive) probing point process. Hence, the Palm probability is simply the average fraction of probes in $(a, b]$ which observe the sample path of $Z(.)$ as belonging to A. Note that the Palm probability does not depend on a, b.

We can immediately establish that when probes and cross-traffic are independent, the Palm probability and the underlying probabilities are equal.

$$\begin{aligned}\mathcal{P}_0(A) &= \frac{1}{\lambda(b-a)} E\Big[\sum_{T_n \in (a,b]} 1_{Z(T_n)\text{satisfies}A}\Big] \\ &= \frac{1}{\lambda(b-a)} E\Big[\sum_{T_n \in (a,b]} 1_{Z(0)\text{satisfies}A}\Big] \\ &= P[\text{A satisfied}] \frac{1}{\lambda(b-a)} E\Big[\sum_{T_n \in (a,b]} 1\Big] = P(A). \end{aligned} \tag{4.5}$$

The first reduction is due to stationarity and independence, and the second to independence. This result implies that the average over all sample paths of what an independent probe streams sees (represented by $\mathcal{P}_0$) is equal to the 'ground truth' (represented by $P(A)$). As we discussed in Section 4.2.4, this lack of bias does *not* mean that every instance of a probing stream is good for our purposes. In fact, for (almost) every sample path to be good we need the ergodicity property which is the asymptotic convergence of Equation 4.4.

4.3.3 NIJEASTA and NIMASTA

We explained the need for joint ergodicity as a way of avoiding possible 'phase-locking' between the probes and cross-traffic. We now state three key theorems from Palm calculus [BB03] that build upon this intuition and lead toward determining when sampling bias is zero.

So far, we defined joint ergodicity with respect to (w.r.t.) the underlying system product space probability P. Analogous definitions can be made by replacing P with the Palm probability $\mathcal{P}_0$. Note that ergodicity w.r.t. $\mathcal{P}_0$ is defined using just a single shift θ_{T_1}. Since under $\mathcal{P}_0$ the first point lies at $T_0 = 0$, this *discrete shift* θ_{T_1} has the effect of simply shifting the origin to the next point. The same shift can be used recursively to move down the probe sequence.

Ergodicity w.r.t. P is quite different from ergodicity w.r.t. $\mathcal{P}_0$. The former says that empirical averages converge for an observer who can access the $Z(t)$ sample path continuously over time.

In contrast, ergodicity w.r.t. the Palm probability means that empirical averages converge for an observer who can access only the samples $Z(T_n)$ for all n. Though ergodicity w.r.t P and $\mathcal{P}_0$ represent different properties, the following theorem from [BB03] (Property 1.6.3, pp.52) states that they are equivalent.

Theorem 1 *θ_t-ergodicity with respect to P is equivalent to θ_{T_1}-ergodicity with respect to $\mathcal{P}_0$.*

Our second theorem, proved in [BB03] (Theorem 1.6.1, pp.47), formalizes our intuition that asymptotic convergence (Equation 4.4) is satisfied if the product space is ergodic w.r.t. $\mathcal{P}_0$.

Theorem 2 ***Discrete Pointwise Ergodic Theorem:*** *Assume that $(\mathcal{F}, \mathcal{P}_0)$ is ergodic w.r.t. the shift θ_{T_1}. Then, for all positive functions f, the following holds*

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n)) = E_0[f(Z(0))] = \int f(Z(0))\mathcal{P}_0(dZ).$$

Finally we have our main result, a very general condition under which Equation 4.4 holds.

Theorem 3 ***NIJEASTA:*** *If the product space is ergodic w.r.t. the Palm probability and the probing stream is independent of cross-traffic, then asymptotic convergence holds.*

Proof: By assumption the discrete pointwise ergodic theorem applies, and since the probing and cross-traffic processes are independent, by Equation (4.5) P can be replaced by $\mathcal{P}_0$:

$$\begin{aligned}\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n)) &= \int f(Z(0))\mathcal{P}_0(dZ) \\ &= \int f(Z(0))P(dZ) = E[f(Z(0))]. \ \square\end{aligned}$$

Thus, asymptotic convergence is guaranteed with an independent probing stream as long as the product shift is ergodic w.r.t. $\mathcal{P}_0$, or equivalently (using Theorem 1) P. This result can be summarized as:

NIJEASTA:

Non-Intrusive Jointly Ergodic Arrivals See Time Averages.

The jointly ergodic assumption of NIJEASTA is similar to the Lack of Anticipation Assumption of

Wolff in that it states exactly what is required, but does not say when it is true, which can be inconvenient in practice. Our last theorem is classical ([Pet83], Theorem 6.1, pp.65) and states simple sufficient conditions under which the joint ergodicity holds, based on the idea of *mixing*. A point process $\mathcal{P}$ (or equivalently its shift) is said to be mixing[3] if, for all events A, B:

$$\lim_{t\to\infty} P_{\mathcal{P}}(A \cap \theta_{-t}(B)) = P_{\mathcal{P}}(A)P_{\mathcal{P}}(B).$$

Intuitively, mixing is a special (and strong) form of ergodicity where on separation under the shift, all memory between any sets A and B is lost, so that they ultimately act as independent events.

Theorem 4 *The product space $\mathcal{F}$ of $\mathcal{P}$ and $\mathcal{T}$ is ergodic whenever at least one of them is a mixing process, and the other ergodic.*

Of the two cases covered in this theorem, that of a mixing probe process has practical importance, because although we may suspect that cross-traffic is mixing, say in the Internet backbone where myriads of random effects wash out deterministic synchronization, we cannot guarantee it. On the other hand, if we choose to always use probing processes which are mixing, we are assured of satisfying the joint ergodicity conditions required for zero sampling bias, regardless of the dynamics of cross-traffic. To highlight this property, which generalizes PASTA (in the non-intrusive case only!), we coin:

NIMASTA:

Non-Intrusive Mixing Arrivals See Time Averages.

It is useful to review at this point the observations of Section 4.2. Three kinds of processes appeared there: stationary renewal processes (with exponential, uniform, or Pareto inter-arrivals), the periodic process (with random phase), and the EAR(1) process. As is well known [DVJ88]. renewal processes are mixing provided that the support of the inter-arrival distribution contains an interval, and the EAR(1) process is also strongly mixing [GL80]. However, the periodic process is not, although it is (by itself) ergodic. The non-intrusive examples throughout Section 4.2, in particular in Section 4.2.4, illustrate NIMASTA and NIJEASTA at work, depending on whether cross-traffic and/or probe traffic is mixing or not.

[3] In fact, both *weak* and *strong* mixing can be defined [Pet83]

4.3.4 From Delay to Jitter

So far we have considered positive functions f which act on Z at a single time point only. In fact more general functions of the form $f(Z(0), Z(t_1), \ldots, Z(t_k))$ can be considered, which gives access to the temporal structure of Z. Key examples are the n-dimensional distributions of the process, and the *delay variation* or *jitter*.

Palm Calculus can deal with this greater generality by considering clusters of (non-intrusive) probes sent at ergodic times times $\{T_n\}$. Each cluster consists of $k+1$ probes sent at times $T_n + t_i$, $i = 0, \ldots k$ with $T_0 = 0$. Palm calculus can then be applied by formulating the clusters as marks, the probe process thereby becoming a *marked* point process (for details see [BB03]). As before, one can measure the average behavior of any such function without bias. Formally,

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n), \ldots Z(T_n + t_k)) = E[f(Z(0), \ldots Z(t_k))] \tag{4.6}$$

As an example, we show how to measure jitter on a time scale of τ, that is, we desire the distribution of $J_\tau(t) = Z(t+\tau) - Z(t)$. Let the clusters arrive as a renewal process with inter-arrival distributed uniformly over $[9\tau, 10\tau]$. This process is mixing. Each cluster will consist of two points, the cluster seed at T_n, and a trailing probe at $T_n + \tau$. We then simply collect the jitter values $\{J_\tau(T_n)\}$ and estimate its distribution by forming a histogram (technically, this implies defining multiple functions f, each an indicator function for a histogram bin, and counting the hits in each. These counts are always positive as required, although jitter itself can take either sign).

4.4 Experiments

In this section, we illustrate the non-intrusive results that we have developed so far using simulation experiments. We used *ns-2* [Sim] to conduct our simulations. In Section 4.2, we demonstrated our results using simple single-hop simulations. Our goal, in this section, is to present realistic scenarios that illustrate the universal applicability of NIMASTA. We first show the importance of NIMASTA by illustrating a carefully constructed example in which periodic probing streams are biased. We use window-constrained TCP and periodic UDP streams in this example. Then, we show that NIMASTA is valid when cross-traffic has feedback. We use long-lived TCP flows that

saturate the path to show this. We then use a suitable definition of "jitter" to show that our extension to NIMASTA (see Section 4.3.4) is valid too. Finally, we simulate a much more complex network with both short-lived web traffic and persistent traffic, i.e., cross-traffic that flows multiple links of the path.

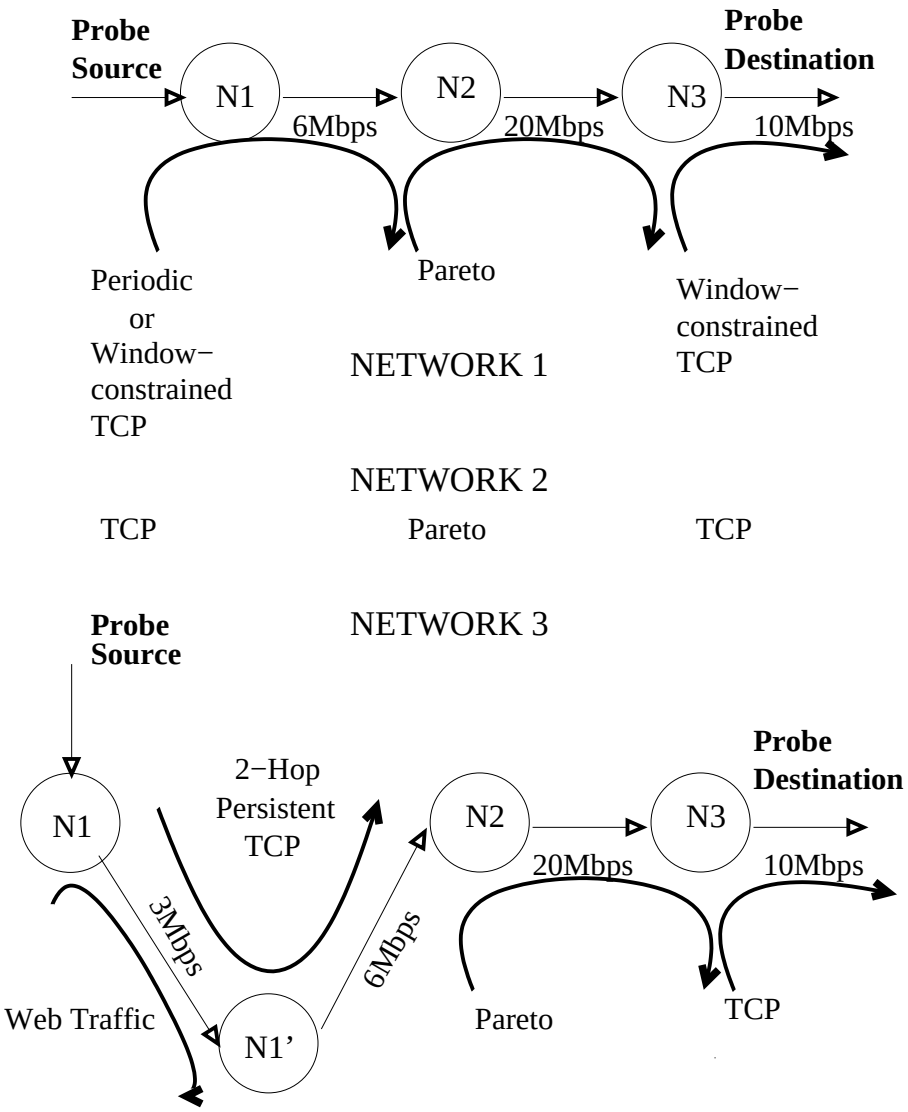

Figure 4.8. The networks that we used in our simulation experiments.

We used the networks shown in Figure 4.8. Notice that all of them have at least 3 hops and are listed in increasing order of the the complexity of their cross-traffic. We used Pareto cross-traffic and higher link capacity for the middle hop as a way of simulating the high-speed portion of many Internet paths. We simulated these networks for more than 100 seconds. We used probe streams

with arrival processes that are periodic, Poisson, uniformly distributed and distributed according to a Pareto distribution. Recall that, except for periodic probing, the rest are mixing. For all of them, we used a mean inter-arrival time of 10ms. Consistent with the rest of this section, we achieved non-intrusive probing by using zero-sized probe packets. We compared the probe-derived estimates with the ground truth obtained using the *Ground Truth Calculator (GTC)* described in Chapter 1. Recall that GTC used arrival and departure logs at all hops to calculate the ground truth. As the time resolution of ns-2 is 1μs and packet sizes are an integer number of bytes, we could use link capacities of at most 20Mbps. Larger link capacities caused significant rounding-off errors.

4.4.1 Periodic Effects

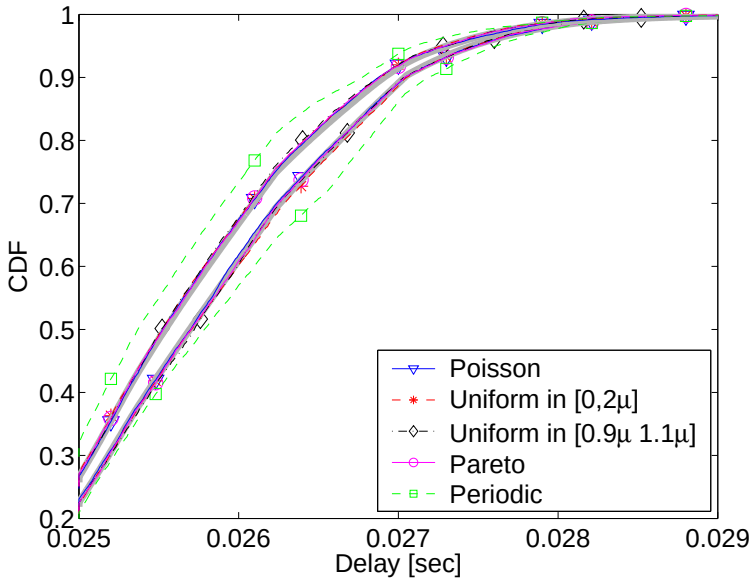

Figure 4.9. Simulation showing the validity of NIMASTA in a multi-hop system, and sampling bias due to phase-locking. The estimated CDFs are plotted with a large (10000) number of probes. Left set of curves: periodic cross-traffic on hop 1, Right: window-constrained TCP flow on hop 1.

We start with Network 1 shown in Figure 4.8. Two sets of results are given in Figure 4.9, depending on whether the cross-traffic on the first hop is periodic, or window-constrained TCP. In each case the delay marginals show that NIMASTA holds for each of the mixing probe traffic. But, for the periodic probes, the probe traffic and cross-traffic become "phase locked". We achieve the phase-locking by choosing the probing period to be commensurate with the round-trip time of the window-constrained TCP flow on the first hop (right set of curves), and as a multiple of the

periodicity of the UDP cross-traffic (left set of curves). This illustrates that, synchronization effects in the network can cause periodic behavior. This periodic behavior can, in turn, cause periodic probing to be biased.

4.4.2 TCP-saturated links

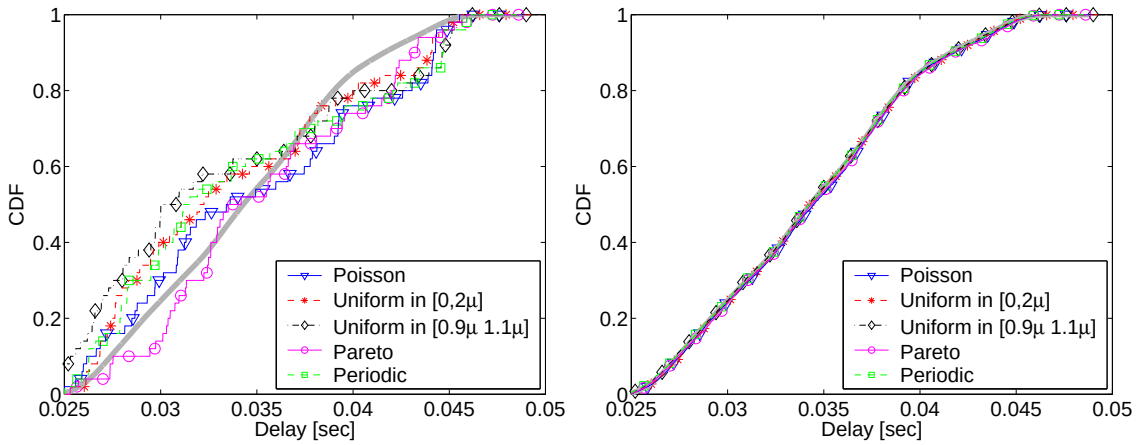

Figure 4.10. Simulation results showing the validity of NIMASTA with saturating long-lived TCP flows. The left plot shows the estimated delay distribution with a small number of probes (100) and the right plot shows the estimated distribution with a large number of probes (10000).

Next, we use Network 2 that replaces the window-constrained TCP flows with long-lived saturating TCP flows. This causes 100% utilization on the first and third links and allows us to investigate if NIMASTA is valid when cross-traffic uses feedback. We plot the results in Figure 4.10. We show two plots. The left plot shows estimates obtained with 100 probes whereas the right plot shows estimates obtained with 10000 probes. This illustrates two things. First, all of them converge asymptotically. Second, with a small number of probes, variance is significant.

4.4.3 Jitter

Our next set of experiments use Network 2 to illustrate the extension to NIMASTA (see Section 4.3.4) that considers multi-dimensional functions of end-to-end delay. Specifically, we consider the difference in delays of two packets sent 1ms apart. We call this difference "jitter". To calculate the ground truth jitter, we first use GTC to calculate the ground truth virtual delay distribution. The ground truth of jitter is calculated as the difference between this ground truth virtual delay and a

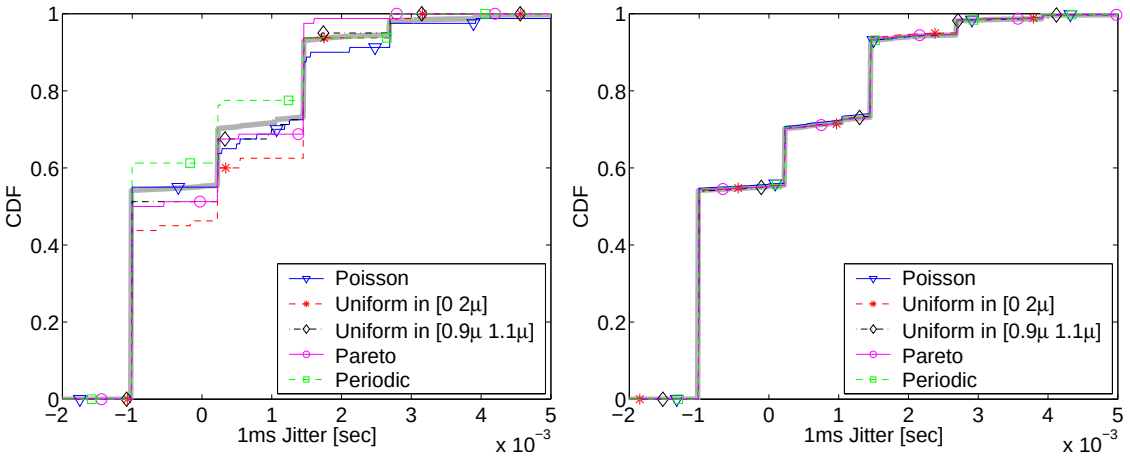

Figure 4.11. Simulation results showing the validity of NIMASTA with multi-dimensional delay functions. We plot the estimated and ground truth distribution of jitter, the difference in delays of two zero-sized packets sent 1ms apart. The left plot shows the estimated CDF with few probes and the right plot shows the estimated CDF with a large number of probes.

time-shifted version of it. We compare the ground truth jitter distribution and estimated jitter in Figure 4.11. To estimate jitter, we sent pairs of (virtual) probes according to the chosen arrival process. For instance, the time between the closest probes of two consecutive pairs of Poisson probing was exponentially distributed. We chose this to avoid overlap between the probe pairs. As can be seen from Figure 4.11, our extension to NIMASTA clearly holds.

4.4.4 Short-lived and Persistent Cross-traffic

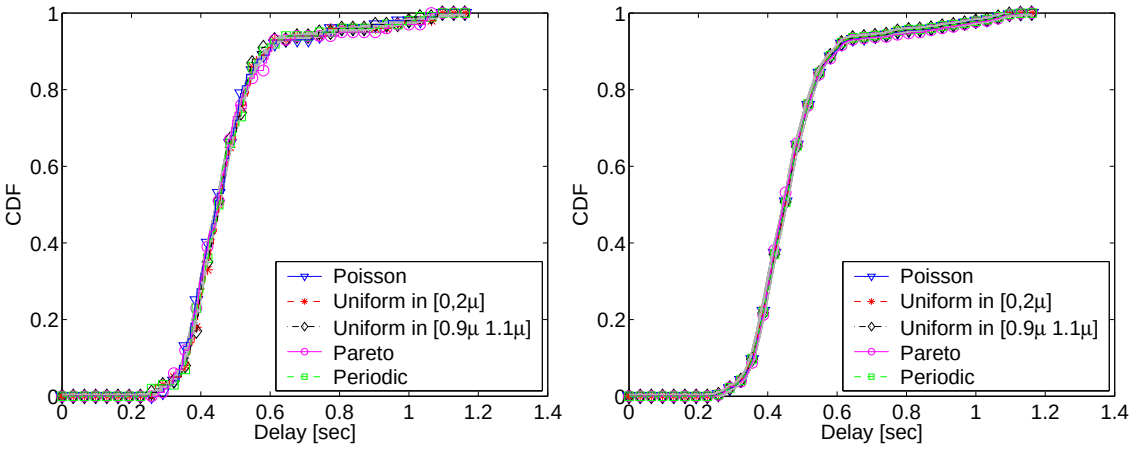

Figure 4.12. Simulation results showing the validity of NIMASTA with a complex network that has persistent cross-traffic, cross-traffic with feedback (TCP) and realistic cross-traffic (web traffic).

Finally, we introduce much more complexity by simulating Network 3 of Figure 4.8. First, we add an additional hop of $3Mbps$ to the first hop. Then, the saturating TCP flow is made 2-hop persistent - it traverses the first two hops that the probes traverse, too. In addition, we introduce web traffic that shares the first hop. We generated web traffic using the example provided with *ns-2*[Sim]. This example uses 420 web clients and 40 web servers. We plot the results in Figure 4.12. To obtain these results, we faced a difficulty. We found that the ground truth virtual delay distribution actually changed based on the seed of the simulation. This seems to indicate an interesting point - TCP congestion control and short-lived flows might cause a large variance in the ground truth itself. Since addressing this is beyond our scope, we modified our simulations so that all five probing streams were simultaneously active in one simulation. This ensured that the streams were measuring the same ground truth. Our results in Figure 4.12 further confirm NIMASTA. Notice that the absolute delay was of the order of a second. In comparison to this, the variance of the CDFs estimated with few probes is small.

4.5 Intrusive Measurement

The last section dealt with the non-intrusive case, i.e., when probes are of zero size. Here we consider implications for inversion and sampling bias arising from 'real' probes of positive size.

4.5.1 Setting

The general setting of Section 4.3.1 continues to hold, with the key difference that now probes influence system evolution. This does not affect the existence of the virtual delay process $W(t)$ (what a zero sized observer would see), nor our final aim, namely to measure $Z^*(t)$, the delay probes would have observed had they arrived to the *unperturbed* system (if probes were not present). However there are two important changes:

- Our observable $Z(t)$ can no longer be taken to be $W(t)$, as we use positive-sized probes. Instead, the probe delays are observables for a "ground truth" that depends on the probe

packet size p. This "ground truth" corresponds to the delay that a packet of size p would observe when injected into the system at steady state.

- Observations of $Z(t)$ are not observations of $Z^*(t)$: an inversion issue arises.

Our goal is to estimate $Z^*(t)$ using observations of $Z(t)$ at the probing times T_n, and to determine in what sense, if at all, the following modified form of Equation 4.4 holds:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n)) = E[f(Z^*(0))]. \tag{4.7}$$

Notice that this equation defaults to Equation 4.4 if probes are non-intrusive, in which case $Z(t) = Z^*(t) = W(t)$.

4.5.2 PASTA

In the intrusive case, probe samples $Z(T_n)$ may be 'anticipated' by the system, resulting in sampling bias. For instance, consider the uniform renewal process with support on $[0.9\mu, 1.1\mu]$ of the left hand plot in Figure 4.5. The negative bias results from the probes only weakly seeing the contribution to load of other probes, which arrive at least 0.9μ from them.

Sampling bias due to intrusiveness means that Equation 4.5 does not in general hold (at least for independent probing streams), but in the Poisson case it can be replaced with:

Theorem 5 ***PASTA****: For Poisson probes (intrusive or not) the "memoryless" property of the exponential inter-arrivals implies*

$$\mathcal{P}_0(A) = P(A). \tag{4.8}$$

Further, since a Poisson process is mixing, Theorem 4 is true and so the product shift is ergodic. Therefore, the same set of reductions as in Section 4.3.3 can be applied:

$$\mathcal{P}_0(A) = P(A) \implies E_0[f(Z(0))] = E[f(Z(0))] \tag{4.9}$$

$$\implies \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(Z(T_n)) = E[f(Z(0))] \tag{4.10}$$

This statement of PASTA in our setting reaffirms the fact that what it provides is unbiased sampling of (functions of) the total system Z. This says nothing about Equation 4.7, which includes the inversion step taking us back to our target, Z^*.

Experiments

Figure 4.13 illustrates PASTA using simulations with Network 1 of Figure 4.8. We used the same experimental setup as in Section 4.3.3 with one difference. The probe packets had a constant non-zero size p. To compare the delays observed by these packets, the ground truth was the delay that a packet of size p injected at time t would experience. Again, we used our *Ground Truth Calculator (GTC)* to calculate this.

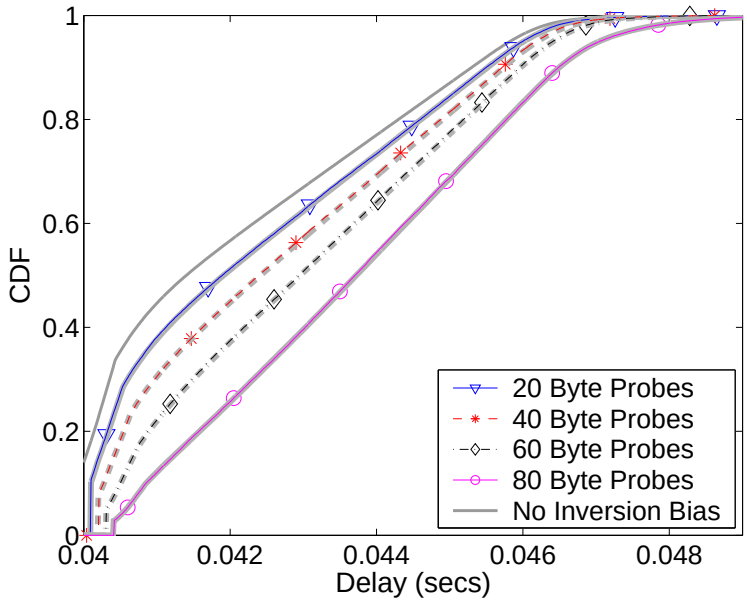

Figure 4.13. Simulation showing the validity of PASTA in a multi-hop system, albeit with inversion bias, for 4 different packet sizes (intrusiveness levels).

Delay marginals, obtained from 50000 probes, are plotted over a range of intrusiveness, achieved with 4 different probe sizes in Figure 4.13. We used the large number of probes because our goal was to verify PASTA which is an asymptotic result. The figure shows, as we expect, that PASTA continues to hold for delay, despite the dangerous periodic components of cross traffic. Notice that, the "ground truth" in each of the intrusive experiments is calculated as that corresponding to the probe size used in that experiment. We also plot the CDF of $W(t)$, the virtual delay process in the unperturbed system to illustrate the problem of inversion bias. Inversion bias exists even w.r.t the ground truth of any positive-sized packet in the unperturbed system too (not shown in the figure).

4.5.3 Minimizing Intrusiveness

Conventional wisdom holds that Poisson streams should be used for active network measurements due to PASTA. However, as discussed above, Poisson streams measure the perturbed system and offer no guidance on inversion. Hence, in the absence of universally applicable inversion methods, PASTA alone is of little use. One strategy is to use (e.g. [ZDP01]) a minimally perturbative stream to reduce inversion bias. Since packet sizes cannot be made arbitrarily small, non-intrusiveness is achieved by 'rare probing', i.e., making the inter-arrival times large.

Any argument to justify minimal inversion bias due to the "almost non-intrusiveness" of the probing stream can justify minimal sampling bias too. Specifically, if the inversion bias is considered negligible due to rare probing, then each probe does not see the effect of previous probes. For practical purposes, the probing is non-intrusive and by NIMASTA, any mixing process can be used. PASTA is irrelevant because we are eliminating the advantage Poisson probes have over other mixing probes - lack of "memory" about themselves. In fact, the following generalization of NIMASTA is the desired result.

Rare Probing: Denote $\mathcal{P}_a$ to be a dilation of the probing process, i.e., $\mathcal{P}_a$ refers to the probing process with arrivals at $\{aT_n\}$. Denote the delay in such a system to be $Z_a(t)$. Then, we want

$$\lim_{a\to\infty} E_0^a(f(W^a(0))) = E(f(W^*(0))). \tag{4.11}$$

Notice that the above limit implies that both inversion and sampling bias go to zero in this limit. A proof of the above statement for a general class of systems, which allow a Markov state representation, is provided in [BMVB06]. While this disallows systems with long-range dependent cross-traffic arrivals, we believe that similar results do hold, at least for most practical systems. Figure 4.13 empirically shows this result, too.

Poisson probing has been suggested for packet pairs [SKK03], and packet trains [LRLL04]. However, there is no theoretical rationale for this. PASTA applies only to a stream of Poisson *packets* that measure *delays*, and cannot justify any inference based on temporal behavior between probe of a pair, where interactions are not weak or memoryless. In fact, rare probing requires that packet trains be well separated, so Poisson probing of arbitrary rate (as we saw in section Section 4.2.3), is likely to be suboptimal.

The use of rare probes has four advantages. First, it provides access to delays seen by a non-zero sized packet. Second, it avoids the problem that in fact the Poisson process is physically unrealizable. Third, probe arrivals can be chosen based on other properties such as low variance and MSE. Finally, as motivated empirically above, the rare probing approach would not be restricted to single probes, but *could include arbitrary probe patterns*, provided that the patterns themselves are widely spaced. This gives a rigorous justification of a common practice, where separated groups of non-zero sized probes are sent.

One important question in the rare probing context is "How rare should probing be?". We do not provide a definitive answer to this question. In fact, we believe that the answer is dependent on the specific properties of the path and cross-traffic. In practice, a good rule-of-thumb would be to ensure that a new probe packet be sent only after the previous probe packet leaves the network. Since queues "remember" packets that have left them, this is only necessary and not sufficient. Viewing rare probing as a way to eliminate the need for inversion, a reasonably small probing rate of, say 1%, of the smallest link capacity is a good rule of thumb. This is already satisfied by many existing studies, for example, [ZDP01]. A more sophisticated method would be to use multiple probing rates and use appropriate tests to compare the difference in the resulting estimates. Insignificant differences would indicate the relatively little perturbation that all the probing rates cause and hence, imply acceptable levels of "rarity". The self-consistency principle [Pax04] behind such a method is desirable too.

4.6 Conclusion

Our contributions, in this chapter, have been a rigorous analysis of the exact role and relevance of PASTA in designing estimators of network measurements. Conventional wisdom holds that Poisson sampling must be used in network measurements for avoiding bias, and the PASTA property is cited as the justification for this. In this paper, we use the problem of estimating end-to-end delay to illustrate that this viewpoint is too simplistic and ignores the key aspect of probing - intrusiveness. We show that the zero sampling bias property is shared by a large set of so-called mixing processes when probing is non-intrusive (the NIMASTA property). The zero-bias properties are exclusive to

Poisson only when probes are intrusive. However, it is extremely challenging to invert the perturbed system (observed by probes) to access the unperturbed system. Hence, even with Poisson probes, it is desirable to make them non-intrusive, by sending them as rarely as possible. In such a rare probing context, the choice of the best probing stream is dependent on the total bias (sampling and inversion) and variance. Finally, we show that it is incorrect to use PASTA to justify Poisson packet pairs or trains. Here too, using rare probing at mixing times is a good strategy.

Although we concentrate on active measurements, the general points we make are relevant to other contexts in traffic measurement. For example, one could consider passive end-to-end monitoring, where the delay between packets which are common to two end points are extracted by matching packets from link monitor logs at each end. Alternatively, there are implications for the choice of packet sampling strategies in routers, currently used to reduce the volume of monitored traffic.

Chapter 5

In-Principle Inversion: Cross-traffic Estimation

"It is the theory that decides what can be observed."

- Albert Einstein, German-born physicist

In the previous chapter, we investigated the sampling issues that arise in the context of active network measurements. Specifically, we investigated the question of when to send individual probes, pairs and trains to estimate functions of end-to-end delay and multi-dimensional extensions of delay, e.g., jitter. Our recommendation of "rare probing" was a way to avoid inversion of the perturbed system property to estimate the unperturbed system property. But, often the goal is to access system properties such as available bandwidth [JD03], narrow link capacity [KCL+04]. In such cases, the inversion step has to estimate the desired system property based on observations of some other property, usually end-to-end delay. In this chapter and the next, we tackle one such problem - inverting observed probe delays to access cross-traffic properties. Knowledge of cross-traffic properties is of importance to network operators because it helps them better understand where and why congestion occurs, and how it can be alleviated. Cross-traffic estimation is of interest to end-users, too, given that cross-traffic directly impacts end-user performance metrics such as available bandwidth [SKK03] and delay jitter. Cross-traffic estimators proposed by prior work are not suitable for the Internet due to a variety of reasons. Estimators that assume very specific parametric

cross-traffic models (see [RCR+00, ANT01, SM98]) can be used only if such an assumption is known to be valid. Their advantage, however, is that they provide us access to detailed statistics about cross-traffic. Other cross-traffic estimators [SKK03] make no modeling assumption but only estimate the first-order moment, i.e., the average cross-traffic rate. Moreover, they often require very fine-grained control over packet timings which is very difficult at link speeds that are common in today's high-speed networks. Hence, one of our goals is to achieve the best of both worlds, i.e., access detailed statistics of cross-traffic with minimal modeling *and* without requiring fine-grained control over packet sending times.

Our focus, in this chapter, is on in-principle inversion methods to access cross-traffic statistics. In the next chapter, we use these methods to design practical cross-traffic estimators. Throughout, we choose to use the delays of probe packet pairs as the observed path property. Choosing pairs is natural since the delays of a pair or train are much more likely to capture the nature of intervening cross-traffic than individual probe delays. Pairs are chosen over probe trains because trains only complicate analysis without providing any new insights. Additionally, we assume that the First-In First-Out (FIFO) scheduling discipline is used. This is the most commonly used scheduling discipline in today's routers and was also recently validated [HVPD04]. We start our analysis by investigating *what* can be estimated in the case of a single (FIFO-based) hop. Using basic queueing theory we show that probe pair delays expose two cross-traffic functionals representing average rate and burstiness. Then, we show that sample path ambiguity arises, i.e., the same observations of delays can be caused by entirely different cross-traffic arrivals, unless we make some assumption on cross-traffic. Therefore, we assume that the delay of the first probe of a pair is independent of the cross-traffic trapped between the pair. This is quite general and well-motivated, as we show in the next chapter using real data. Moreover, this assumption makes inversion tractable by allowing us to access the CDFs and joint density of the two functionals *almost* entirely. Using an intuitive, geometric framework, we develop exact inversion expressions and approximate expressions that prove to be more useful in practice. We refer to the former as Class 1 and the latter as Class 2 expressions. Finally, we investigate adapting our theory to analyze multi-hop scenarios. We find that such multi-hop analysis is encumbered by two fundamental effects - the unknown persistence

properties of cross-traffic, i.e., how many consecutive hops does cross-traffic flow across, and the unobservable times at which probe traffic arrives at the intermediate hops.

This chapter is structured as follows. We start by discussing, in Section 5.1, prior works related to cross-traffic estimation. We use this discussion to motivate our target problem. We also provide an overview of our contributions in this chapter. In Section 5.2, we derive the two cross-traffic functionals that are exposed by probe pair delays in the case of a single (FIFO) hop. We also illustrate how sample path ambiguity can arise if we make no assumption about the cross-traffic. In Section 5.3, we derive (exact) Class 1 inversion expressions under our well-motivated assumption on cross-traffic. We use an intuitive, geometric framework to derive these expressions. In Section 5.4, we develop more useful (approximate) Class 2 inversion expressions. In the next chapter, we not only validate the aforementioned assumption but also show that estimators based these Class 2 expressions work well in practice. In Section 5.5, we investigate if the developed theory can be adapted to the multi-hop case. We find that, it is hard to do so due to quantities that are not observable and unknown cross-traffic persistence properties. We summarize in Section 5.6.

5.1 Overview

In this section, we first survey prior work related to cross-traffic estimation and discuss their shortcomings. Then, we state our target problem and elaborate on the scope of our investigation. Finally, we present an overview of our contributions.

5.1.1 Motivation - Bandwidth Inversion Methods

In his work on congestion control, Jacobson [Jac88] first observed that inter-packet separation is preserved after the slowest link along a path. Keshav [Kes95] proposed using this to estimate the bandwidth available to a flow in a network employing round-robin schedulers. Then, Carter, et al. [CC96] applied these ideas to FIFO-based network paths. They proposed methods to estimate (capacity and) available bandwidth. Later, Melander, et al. [MBG00] proposed more heuristic methods to estimate available bandwidth on a FIFO-based path. Their tool, TOPP, used the input and output

rates of packet pairs. Other techniques (see [JD03], [HS03]) extended these ideas and used packet trains to estimate available bandwidth. The underlying principle behind all of these is that available bandwidth is the probing rate at which the bottleneck link is saturated. These techniques determine that the bottleneck link is saturated by observing the resulting increases in delay. For accurate estimation, the delay increases must be significant. Hence, they are fundamentally intrusive. These ideas on causing queue buildups have been adapted to estimate the congested hops along a path [ASM03, HLM$^+$04]. However, these techniques cannot quantify the available bandwidth on these hops [HLM$^+$04].

All of the above methods measure available bandwidth and do not directly measure cross-traffic. Techniques performing direct estimation of cross-traffic are also known. We discussed these previously in Chapter 2. Below, we discuss them again to re-emphasize their shortcomings. These techniques use probe delays to measure the amount of cross-traffic "trapped" between them. An example of such a technique is Spruce proposed by Strauss, et al. [SKK03]. They used a simple one-hop model to observe that, with intra-pair separation times less than the transmission time of the first packet, we can obtain the cross-traffic rate using the input and output gap of the pair. Techniques such as Spruce [SKK03] use large packets to ensure that the queue remains busy between the packet pair. Ribeiro, et al. [RCR$^+$00] also assumed that the queue is busy between a pair. They used this to estimate the amount of intervening cross-traffic and estimated cross-traffic properties assuming a parametric multi-fractal model. The requirement that the queue be busy between the two probes can often not be ensured. For instance, on a 100Mbps link the separation of two maximum-sized packets (1500 bytes) should be less than $120\ \mu s$. To estimate available bandwidth, these methods subtract the (estimated) average cross-traffic rate from capacity estimates. Note that packet pairs have also been used in estimating capacities, e.g., Capprobe [KCL$^+$04]. Such capacity estimation methods use minimum filtering to pick out the pair that saw no cross-traffic and use this to estimate the bottleneck capacity.

A few prior works proposed estimators of cross-traffic assuming that it arrived according to a specific parametric model. For instance, Ribeiro, et al. [RCR$^+$00] assumed that cross-traffic is multi-fractal. Sharma, et al. [SM98] proposed estimators of average cross-traffic rate assuming Poisson cross-traffic. Alouf, et al. [ANT01] too assumed a Poisson arrival process to estimate

cross-traffic and queue sizes in the single-hop case. The assumption of a very specific model limits the use of such techniques.

Recent work by Liu, et al. [LRL05b] investigated the signals that packet pair and train delays expose. But, their results are asymptotic in nature and mostly help understand inaccuracies that may arise with the fundamentally intrusive techniques, e.g., *pathload* [JD03]. These inaccuracies are caused because the discrete nature of data traffic can cause queue buildups even when the bottleneck link is not saturated. Subsequently, Liu, et al. [LRL05a] focused on a standard assumption, made by most of the above prior works, that a path can be modeled as a single hop representing its predominant bottleneck. This model, while not accurate, is a reasonable approximation for many Internet paths. The analysis in [LRL05a] showed that, for fundamentally intrusive techniques such as *pathload*, there is a trade-off between accuracy and intrusiveness. Specifically, the longer a packet train is, the more is its robustness to queueing on other hops. Nevertheless, such techniques are more robust to queueing at other hops compared with non-intrusive packet pair techniques, e.g., Spruce [SKK03]. This is because, non-intrusive techniques approximate the input and output gap at the bottleneck with the sending and receiving gap. Even moderate amounts of queueing at other hops can have a huge impact given that the input and output gaps are typically very small. For purposes of available bandwidth, errors due to the necessary capacity estimation can also lead to errors with techniques such as Spruce. However, the non-intrusiveness of Spruce and similar techniques involving probe pairs or small probe trains makes them ideal for use on distributed infrastructures such as Planetlab [Pla] which limit user data rates.

5.1.2 Scope of Investigation

The above discussion shows the lack of satisfying answers to two questions. The first is whether we can design non-intrusive techniques for cross-traffic estimation that do not require fine timing control. Such techniques are naturally of interest in measuring paths with a single predominant bottleneck - a common-case scenario on the Internet. The second question is whether we can extend such techniques to consider truly multi-hop paths. Our work, in this chapter and the next, answers the first question by designing cross-traffic estimators for a single hop. We briefly touch upon the second question in this chapter by motivating why it might be impossible to extend our estima-

tors to a multi-hop path. Later, in Chapter 7, we propose a novel measurement-friendly network architecture that can be used to apply single-hop methods including our estimators to multi-hop paths.

In investigating the problem of cross-traffic estimation, we choose to use the delays of pairs of probe packets as our input. Intuitively, pairs are better that individual probes since their delays are related to each other via the cross-traffic "trapped" between them. We choose pairs over trains not only because they are easier to analyze but also because such analysis can arguably be generalized to packet trains. Hence, the following is our target problem:

Given a knowledge of the measured delays of probe pairs, what can be learned about the probability laws governing the cross-traffic?

Our objective is to answer the above problem in a rigorous manner. Hence, our first sub-goal is to investigate what cross-traffic properties we *can* learn and what properties we *cannot*, in theory. Our second sub-goal is to design in-principle inversion expressions to access cross-traffic properties. The pros and cons of the few existing cross-traffic estimators, discussed earlier, dictate the attributes that we desire our inversion expressions to possess. In particular, we want to perform minimal modeling, yet access detailed statistics. Additionally, we want to perform our analysis without assuming an always-busy queue (between the pair) and/or stipulating the fine-grained timing required to achieve an always-busy queue. Our third sub-goal is the actual design and evaluation of practical estimators. This is addressed in Chapter 6.

We use the prevailing hop model consisting of a FIFO queue to which both cross-traffic packets and probes effectively arrive instantaneously, but flow out deterministically as they are serialized onto the output link. This abstraction of hop behavior is appropriate in today's Internet where store and forward router architectures are common, with fast switch fabrics where through-router delays are concentrated in output buffers. Recently, Hohn, et al. [HVPD04] validated this abstraction using real data collected at the input and output interfaces of a router in the backbone network of a tier-1 ISP.

5.1.3 Contributions

First, we develop a packet-based model of a FIFO queue. Our model is based on Lindley's equation [BB03] from queueing theory. Bolot [Bol93] used this equation in his work though many later works [HS03] only considered a simpler fluid-flow model. Our model also demonstrates that two functionals of cross-traffic, measuring the average amount and burstiness, relate probe pair delays. These are, therefore, the only properties of cross-traffic that can be derived from observing probe pair delays.

To determine whether the cross-traffic functionals can be derived from probe pair delays, we perform a very simple sample path analysis. This shows that the non-linearity of the queue (its size does not go below zero) causes sample path ambiguity, i.e., the same delay observations can be caused by cross-traffic with different properties. The one exception is that, at very small intra-pair times, the amount of trapped cross-traffic is uniquely determined. This effect has been used by previous work [SKK03]; But, as we discussed above, the intra-pair timing is too small to be enforced, especially on a path with other (underutilized) hops.

Sample path ambiguity implies that, to use large intra-pair times, some assumption about cross-traffic is necessary. For applicability in a variety of systems, we assume that the delay of the first probe is independent of the cross-traffic trapped between the pair. Previous work supports this assumption as do we, in Chapter 6. Under this assumption, we show that almost the entire joint distribution of the two functionals can be inverted. We explain inversion by casting conditional probabilities involving the observed probe pair delays and the unobserved cross-traffic functionals into an intuitive, geometric framework. With the help of this framework, we derive inversion expressions for the marginal of the amount of cross-traffic and the joint distribution. We derive exact Class 1 expressions and approximate Class 2 expressions. The latter are more useful in practice because they adapt to available data, as we will see in Chapter 6. Interestingly, we find that a portion of the joint distribution cannot be inverted. Since its size is proportional to the probe sizes, it represents the cost of intrusiveness on observability.

Finally, we explore applying similar theory to analyze multi-hop paths. We find it hard to do so even for a simple two-hop path. We identify two effects - persistence of cross-traffic and

unobservablity of the times at which probe packets reach intermediate hops, as two main reasons why the multi-hop case might be intractable. We use these insights, in Chapter 7, to develop a measurement-friendly architecture that allows non-intrusive single-hop methods usable on multi-hop paths.

5.2 Basics

In this section, we first introduce the system setting we use and notation. Within this setting, we describe the target problem. Then, we use basic queueing theory to explore what properties of cross-traffic are exposed by probe pair delays. We find that two functionals, which can be considered to represent the average amount and burstiness of cross-traffic, relate the delays of a pair of probe packets. These two functionals are not independent; we derive additional expressions involving them, too. Finally, we show that, unless the intra-pair gap is as small as suggested in [SKK03], the system is sample path ambiguous. In other words, different cross-traffic processes can cause the same set of probe pair delays to be observed.

5.2.1 Setting

We consider a single hop of capacity μ that uses a FIFO queue of infinite size. We take the probes to have a constant size of p bytes, which are transmitted in $x = p/\mu$ time units across the hop. Probe pairs are sent at times $\{T_n\}$ with a fixed intra-pair time of t. It is convenient to describe the input traffic in terms of a *random measure* A, whereby the *transmission time* of the cross-traffic arriving to the queue in a time interval I is denoted by the random variable $A(I)$.

Our aim is to recover as much information as possible about the cross-traffic described by A. For this to be feasible, the statistics of the system should not change fundamentally over time, and the probes must be able to collect representative samples of them. The corresponding technical assumptions are that A is stationary, (i.e., for all intervals I, the statistics of $A(\delta + I)$ do not depend on δ) and that the system with probes is jointly ergodic. We showed in Chapter 4 that joint ergodicity is guaranteed by sending these pairs at mixing epochs. The pairs do not need to be rare because our

aim is not to obtain unbiased estimates of the delays that a pair of packets will observe. We only want to sample the cross-traffic process in an unbiased way; This is achieved by sending the probe pairs at mixing epochs. For simplicity, we assume a mixing arrival process that does not cause the pairs to overlap with each other. If we are sufficiently confident of the mixing properties of the cross-traffic, we may use a periodic stream in which every pair of consecutive packets is considered to be a pair (also, see [MVBB05, MVB$^+$05]).

The raw data of a probing experiment are the arrival and departure times of the probe packets. $T_{n,1}$ and $T_{n,2}$ are the arrival times of the first and second packets of the n^{th} pair. Similarly, $T'_{n,1}$ and $T'_{n,2}$ are their respective departure times. We assume no propagation delay and synchronized sending and receiving times for simplifying analysis. In reality, these assumptions are not often justified. But, as we discuss in Chapter 6, what is necessary is *delay variation* [PV02a], and hence, these assumptions are not required in practice.

5.2.2 Two Functionals - Average Rate and Burstiness

To derive how the delays of a packet pair are related to cross-traffic, consider the time period $[T_{n,1}, T_{n,2})$ between the arrival of the packets of the n^{th} pair. If the queue is busy throughout this time period, the departure time of the second probe is simply determined by the delay of the first probe, its transmission time and the cross-traffic $A([T_{n,1}, T_{n,2}))$ between the two probes. If the queue is not busy throughout, then there exists some time instant v after which it is continuously busy (for example, this may be $T_{n,2}$ if the second probe arrives at an empty queue). In such a case, the delay of the second probe is determined only by the cross-traffic that arrives in the time period $[v, T_{n,2})$. Moreover, this delay will be greater than the delay calculated if we had assumed the queue to be busy throughout. We can use these observations to obtain the following version of the well-known Lindley's equation describing FIFO queue evolution (for example, see [BB03])

$$T'_{n,2} = x + \left[\left(T'_{n,1} + A([T_{n,1}, T_{n,2})) \right) \vee \sup_{v \in [T_{n,1}, T_{n,2}]} (v + A([v, T_{n,2})) \right] \tag{5.1}$$

where $x \vee y$ denotes the maximum of x and y. The left hand argument of $\vee$ dominates when the queue is busy throughout the time period between the packets of the pair. If this is not the case, the second term dominates the first and determines the departure time of the second probe. Subtracting

$T_{n,2}$ from both sides of Equation 5.1, we get a relationship involving the delays $D_{n,i} = T'_{n,i} - T_{n,i}$

$$T'_{n,2} - T_{n,2} = x + \left[\left(T'_{n,1} - T_{n,1} + T_{n,1} - T_{n,2} + A([T_{n,1}, T_{n,2})) \right) \vee \sup_{v \in [T_{n,1}, T_{n,2}]} (v - T_{n,2} + A([v, T_{n,2}))) \right]$$

$$D_{n,2} = x + \left[(D_{n,1} + A([T_{n,1}, T_{n,2})) - (T_{n,2} - T_{n,1}))) \vee \sup_{v \in [T_{n,1}, T_{n,2}]} (A([v, T_{n,2})) - (T_{n,2} - v)) \right]$$

Instead of using absolute delay, it is convenient to work with $R_n = D_{n,1} - x \geq 0$ and $S_n = D_{n,2} - x \geq 0$ which are the excess delays above the minimum value of x, the transmission delay of the probes. In terms of the R_n and S_n Equation 5.1 becomes

$$S_n = (x + R_n + C_n) \vee B_n \,, \tag{5.2}$$

where

$$C_n = A([T_{n,1}, T_{n,2})) - (T_{n,2} - T_{n,1}), \tag{5.3}$$

$$B_n = \sup_{v \in [T_{n,1}, T_{n,2}]} (A([v, T_{n,2})) - (T_{n,2} - v)) \,. \tag{5.4}$$

Note that B_n and C_n are functionals of the cross-traffic over the interval $[T_n, T_{n+1})$ *only*, and are neither influenced by the probes nor by the queue state. The following important relationships hold:

1. $B_n \geq 0$ (take $v = T_{n,2}$),
2. $B_n \geq C_n$ (take $v = T_{n,1}$),
3. $B_n \leq C_n + (T_{n,2} - T_{n,1})$ (since $B_n \leq A([T_{n,1}, T_{n,2}))$).

We can interpret C_n as the *net* work that arrives in $[T_{n,1}, T_{n,2})$, and it takes values in $[-(T_{n,2} - T_{n,1}), \infty)$. Thus, C_n gives information on the *integral* of the cross-traffic over a typical intra-pair time period. In contrast, B_n gives some information on the *peak*. More precisely, B_n is the queue size that would have been seen at time $T_{n,2}$ if we considered *only* the cross-traffic arriving in $[T_{n,1}, T_{n,2})$. For example, $B_n - C_n$ is maximized when the traffic arriving over $[T_{n,1}, T_{n,2})$ occurs in a burst just before $T_{n,2}$.

5.2.3 Sample Path Ambiguity

It is clear from Equation (5.2) that we can at most identify the (B_n, C_n) sequence from observations of R_n and S_n. Ideally, we would like to infer the sequence of cross-traffic functionals without

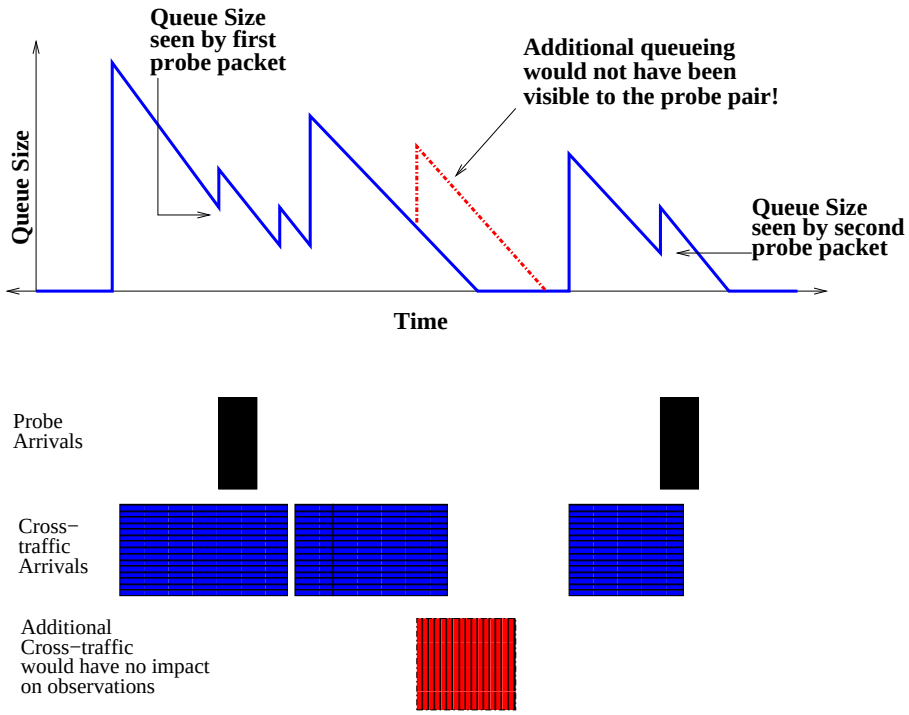

Figure 5.1. A simple illustration of the sample path ambiguity for an arbitrary intra-pair interval. The top plot shows the queue size over time. The dark shaded boxes below it represent the probe packet pair. The x-axis value represents the arrival time and the length of the boxes represents the size of the packets. The second row of packets represent one sample path of cross-traffic packets. The third row shows an additional cross-traffic packet whose arrival would not have changed the observed delays.

making any assumption regarding cross-traffic. However, this is not possible if the queue empties between the pair. We demonstrate this in Figure 5.1. For the same B_n, different values of C_n give rise to the same sequence of observed probe pair delays. In other words, sample path ambiguity occurs as long as modifications of cross-traffic leave unaffected the last busy period. However, if packets of a pair share the same busy period, ambiguity with respect to C_n and A_n, the net and total amount of cross-traffic trapped between them, is eliminated. This is exactly the scenario that techniques such as Spruce [SKK03] hope to exploit. We discussed earlier why this is often not feasible because the intra-pair time t required to achieve such an "always-busy" scenario is very small. For instance, on a 100Mbps Ethernet link, the maximum packet size is 1500 bytes and hence, t cannot be more than $120\mu s$, the transmission time of the maximum packet size.

The existence of sample path ambiguity at the desired t values implies that we need to make some assumption on cross-traffic. In the next section, we discuss what this assumption is and how,

under than assumption, we can invert *almost* the entire joint distribution of the two cross-traffic functionals. For all of our subsequent discussion, we view R_n, S_n, B_n and C_n to be particular instances of respective random variables R, S, B and C. We find it convenient to restate our equations using these random variables and t as the intra-pair time ($T_{n,2} - T_{n,1}$).

$$C = A([0,t)) - t, \tag{5.5}$$

$$B = \sup_{0 \le v \le t} (A([v,t)) - (t-v)) \tag{5.6}$$

$$S = [x + R + C] \vee B. \tag{5.7}$$

Thus, the only information about cross-traffic that we can hope to obtain from packet pair delays are statistics of the above two functionals, B and C (including their joint density). Notice that the marginal distribution of C also gives us the marginal distribution of $A([0,t)) = C + t$.

5.3 In-Principle Inversion

In the previous section, we derived the two cross-traffic functionals exposed by probe pair delays. In this section, we investigate how we to perform inversion in this context, i.e., using the observed delays to access the two cross-traffic functionals. We develop in-principle inversion expressions to derive a variety of statistics of the cross-traffic functionals from observed packet pair delays *irrespective* of t. The statistics that our expressions derive are not restricted to average cross-traffic rate (or available bandwidth), unlike previous work, but include the CDFs and joint densities. We did show in the previous section that such inversion is not possible as long as we do not make some assumption on cross-traffic. We do make such an assumption, namely, that the cross-traffic is such that (B, C) is independent of R. This means that, the delay experienced by the first probe of a pair is independent of the cross-traffic that arrives *after* its arrival and *before* the second probe. We choose this assumption not only because it makes the inversion problem tractable but also because it a reasonable assumption. In fact, in the next chapter, we provide evidence indicating that this assumption is valid for Internet traffic for the time scales of the order of intra-pair time periods. Our performance results in that chapter also bear this out. That this is a reasonable first model of IP traffic over *small* time-scales of a few seconds has been borne out by other studies too [HVA03].

This assumption is much more general than assuming that cross-traffic arrivals follow a specific distribution characterized by a few parameters (as in [RCR$^+$00], for example).

We start this section by introducing an intuitive, geometric framework to explain the observed probe pair delays and the cross-traffic functionals. This framework allows us to derive and understand our inversion expressions. Then, we derive inversion expressions to derive the CDFs of C, the net amount of cross-traffic arriving during an intra-pair time interval. These are exact expressions and we refer to them as Class 1 expressions. Finally, we derive inversion expressions to access the entire joint distribution of the two functionals except for a so-called *ambiguity zone*. We show that the size of the ambiguity zone depends on the transmission time of the probe packets and hence, can be viewed as the cost of the probing intrusiveness.

5.3.1 A Geometric Interpretation

It turns out that the unobservable functionals of cross-traffic and the observable packet pair delays can be viewed in an intuitive, geometric framework. We describe this framework next. Henceforth, we assume that all variables, including time, are discrete. This assumption is not essential, as the discretization can be made as fine as we wish, and in applications using real data, discretization is in any case unavoidable.

Unobservables: Cross-traffic Functional Marginals and Joint Densities

We denote the discrete density and the 2-dimensional cumulative distribution function (CDF) of (B, C) respectively by

$$h(k,l) = P(B = k, C = l), \tag{5.8}$$

$$H(k,l) = P(B \leq k, C \leq l), \tag{5.9}$$

We write $c(l)$ and $C(l)$, $l \geq -t$, for the density and CDF respectively of the marginal corresponding to the variable C, and similarly $b(k)$ and $B(k)$, $k \geq 0$, for B. The relationships listed in Section 5.2 imply that $h(k,l) = 0$ outside of the diagonally oriented "feasible strip" defined by

$$\text{feasible strip:} \quad (k,l) : k - t \leq l \leq k, \quad k \geq 0, \tag{5.10}$$

Figure 5.2 visually shows the feasible strip in the (discrete) two-dimensional space of B and C. Squares with white color have a canonical density of 0. Marginals of C, B, $c(l)$ and $b(k)$, are obtained by summing $h(.,.)$ along infinite horizontal and vertical bars in Figure 5.2. We also find it convenient to define the following "approximations" to $c(l)$ and $b(l)$:

$$c(k,l) = \sum_{i=0}^{k} h(i,l) \tag{5.11}$$

$$b(k,l) = \sum_{i=-t}^{l} h(k,i). \tag{5.12}$$

The sets of k, l pairs in the sums defining $c(\cdot,\cdot)$ and $b(\cdot,\cdot)$ appear in Figure 5.2 as finite horizontal and vertical bars respectively. As is clear from Figure 5.2, $c(k,l) = c(l)$ as soon as $k \geq l + t$.

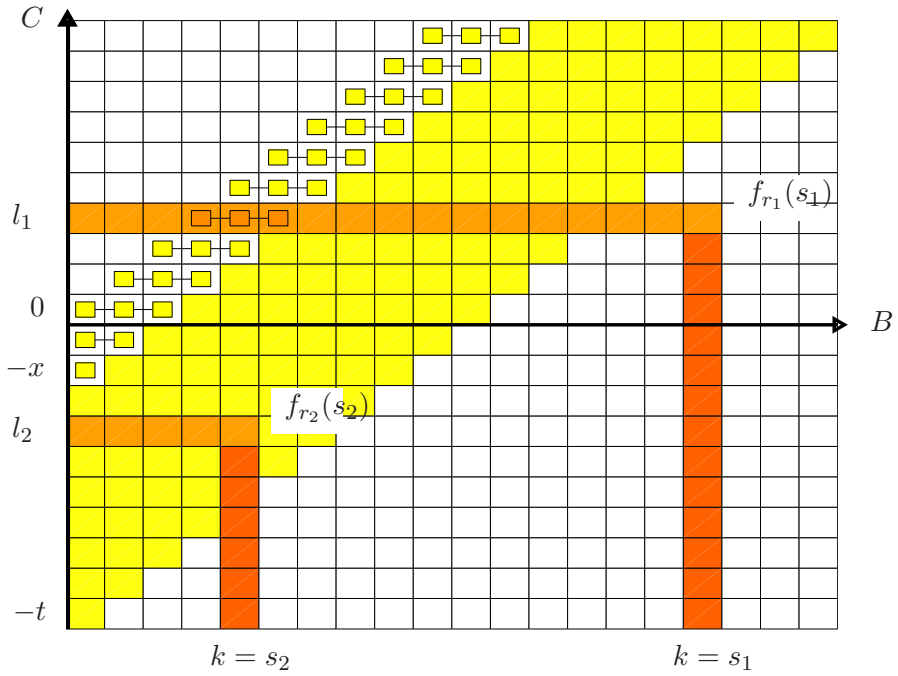

Figure 5.2. The discrete 2-dimensional space of B and C. The domain $\{k, l\}$ where the joint density $h(k,l)$ of (B,C) vanishes is shown as white. The support of (B,C) is the strip $k-t \leq l \leq k$, $k \geq 0$ shown as the light colored band. An observation of $(R,S) = (r,s)$ corresponds to a $(B,C) = (k,l)$ value lying inside an angle shaped set with corner at $(k^*, l^*) = (s, s-r-x)$. Two angle sets are shown (shaded), corresponding to Class 1 (corner outside the strip, $k = s_1$), and Class 2 (corner inside, $k = s_2$). The region where aggregates of $x+1 = 3$ atomic masses are connected is the *ambiguity zone* where individual h values cannot be directly determined.

Observables: Conditional Probabilities

Let $f_r(s) = P(S = s|R = r)$. We can write down this conditional probability by accounting the cases when either of the terms in the right hand side of Equation (5.7) equals s. Using our assumption that R is independent of (B, C), we can count the various possible (B, C) that give rise to $S = s, R = r$ from Equation 5.7 and write

$$\begin{aligned} f_r(s) &= P(B \leq s, C = s - r - x) \\ &\quad + P(B = s, C \leq s - r - x - 1) \qquad (5.13) \\ &= c(s, s - r - x) + b(s, s - r - x - 1) \qquad (5.14) \\ &= H(s, s - r - x) - H(s - 1, s - r - x - 1). \qquad (5.15) \end{aligned}$$

This probability corresponds to a sum of $h(k, l)$ over an "angle shaped" set, with corner at

$$(k^*, l^*) = (s, s - r - x).$$

Two examples of angle sets are illustrated in Figure 5.2. A particular observation of $S = s$ given $R = r$ corresponds to a $(B, C) = (k, l)$ value, which, although unobservable, must lie inside the angle set defined by (r, s). A given (k, l) value may be included in many angle sets corresponding to different (r, s), however the mapping between (r, s) and the corner (k^*, l^*) is linear, and hence uniquely invertible: $(r, s) = (k^* - l^* - x, k^*)$. What are the possible locations of the corners? For a fixed r, as s is increased the corresponding corner values $(k^*, l^*) = (s, s - r - x)$ move upward, tracing out a line parallel to the main diagonal. As r decreases these diagonals translate upward. However, the highest of these, corresponding to $r = 0$, is not the upper boundary of the strip, but lies below it on the line $l = k - x$.

Assuming no constraint on the number of packet pairs, the ergodicity of the system implies that we have access to pairs with $R = r \geq 0$ for all values of r and hence, we can calculate conditional probabilities, $f_r(s)$, involving S and R. Thus, the available information concerning $h(k, l)$ comes in the form of the probabilities, given by $f_r(s)$ for different observed (r, s). All knowledge of the unobservable cross-traffic functionals must be obtained by combining these conditional probabilities (or angle sets) in different combinations.

Next, we derive inversion expressions to access statistics of the cross-traffic functionals using

such conditional probabilities. Our expressions fall into 2 classes. The first class of expressions are exact and provide access to $c(l)$ and $h(k,l)$. Class 1 expressions for the marginal $c(l)$ are based on restricting to cases where the queue is known to be "linear", that is when it is *certain* that the probes share the same busy period. The second class of expressions, derived in the next section, are based on the idea that conditioning on linearity may be too strong to access $c(l)$, and require too much data in practice. Hence, the second class of expressions use suitable approximations of $c(k,l)$ to be $c(l)$, justified by the idea that we can ignore zones where the density is likely to be small.

5.3.2 Exact Inversion Expressions: Class 1

The class 1 inversion expressions that we derive calculate the marginal of C, $c(l)$ and the joint density of (B,C), $h(k,l)$.

Calculating $c(l)$ using a Single Conditional Probability

The first inversion expression uses the observation from Section 5.2 that $B \leq C + t$, which implies that $B \leq x + r + C$ if $R = r \geq t - x$. Hence, for $r \geq t - x$, Equation (5.7) implies that

$$\begin{aligned} P(S = s|R = r) &= P(x + R + C = s|R = r) \\ &= P(C = s - r - x|R = r) \\ &= P(C = s - r - x), \end{aligned} \tag{5.16}$$

which is a function of the *delay variation* $u = s - r$. The last step follows from the independence of R and (B,C) established above. Thus, for each fixed $R = r$ obeying $r \geq t - x$ we have

$$c(l) = f_r(l + r + x). \tag{5.17}$$

In terms of angle sets, the above simply corresponds to taking a corner (k^*, l^*) with $l^* = s - r - x$, and k^* large enough so that the horizontal component of the angle completely traverses the strip ($f_{r_1}(s_1)$ in Figure 5.2). The vertical component in such cases falls below the strip, and thereby contains zero probability.

Calculating $c(l)$ by Combining Multiple Conditional Probabilities

Since the above expression is true for many different r values, it is desirable to combine them, as this would make better use of data in the practical case. A general linear combination can be written as

$$c(l) = \sum_{r=t-x}^{\infty} a_{r-(t-x)} f_r(l + r + x), \tag{5.18}$$

where the a_i are any set of non-negative weights that sum to unity. Intuitively, a good choice is to select weights that reflects the data available: $a_{r-(t-x)} = P(R = r|R \geq t - x)$. It turns out that the resulting expression is the same as if we had set out, looking across different r values, to explicitly collect together all relevant observations with constant delay variation u:

$$\begin{aligned}
P(C = u - x) &= \sum_{r=t-x}^{\infty} f_r(u + r) P(R = r|R \geq t - x) \\
&= \sum_{r=t-x}^{\infty} P(S - R = u|R = r) P(R = r)/P(R \geq t - x) \\
&= \sum_{r=t-x}^{\infty} P(C + x = u, R = r)/P(R \geq t - x) \\
&= \sum_{r=t-x}^{\infty} P(C + x = u, R = r, R \geq t - x)/P(R \geq t - x) \\
&= \sum_{r=t-x}^{\infty} P(C + x = u, R = r|R \geq t - x) \\
&= P(C + x = u|R \geq t - x) = P(S - R = u|R \geq t - x)
\end{aligned}$$

Defining $g_r(u)$ to be $P(S - R = u|R \geq r)$, we get:

$$c(l) = g_{t-x}(l + x). \tag{5.19}$$

The collection of (r, s) values used in this expression is illustrated in Figure 5.3(a), where the shading indicates the corresponding weight. The above expressions essentially choose observations corresponding to large delay values of the first probe in a pair. The large delay ensures that the queue is busy until the next probe arrives. The difference between such a delay and the next delay is used to estimate the distribution of cross-traffic arriving between them.

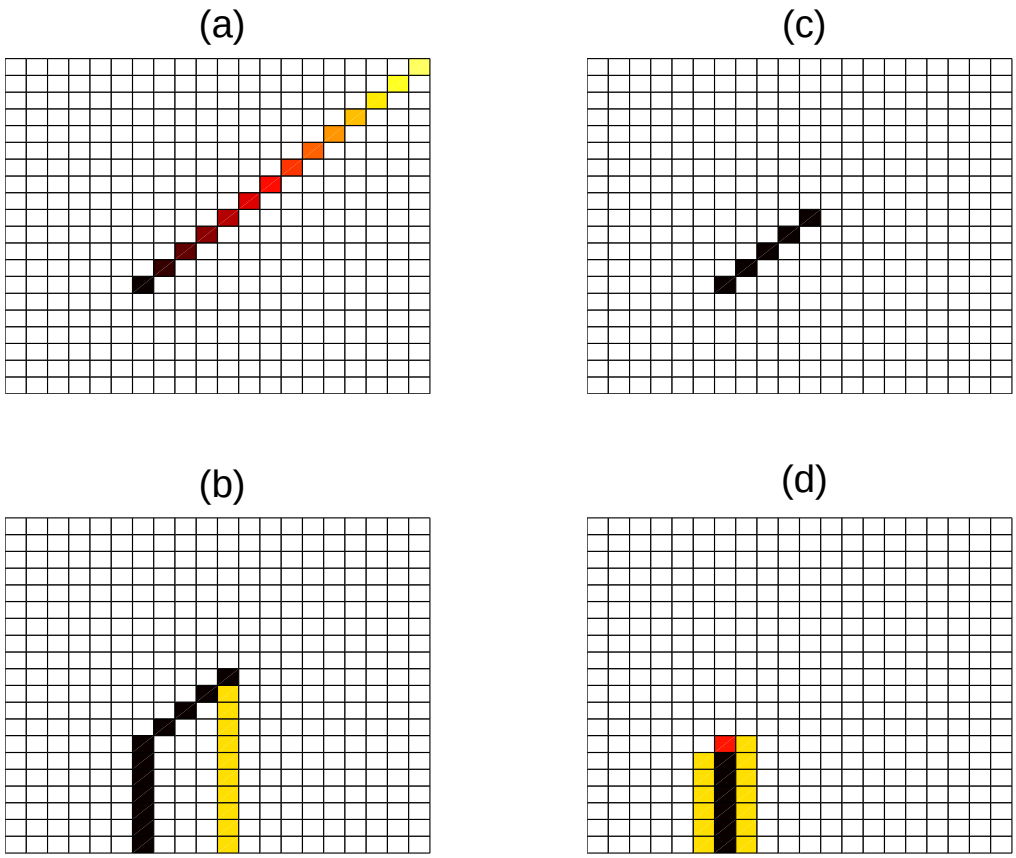

Figure 5.3. The various Class 1 and Class 2 inversion expressions use different portions of the (R, S) space, shown here. Only (d) is an expression to access the joint density. The rest are expressions for the marginal $c(l)$. (a) Equations 5.19 and 5.28 (shading indicates weights used); (b) Equation 5.25 (correction terms give the vertical components); (c) Equation 5.27 (uniform weighting over N values of r); (d) Equation 5.23 (shading indicates term type).

Calculating $h(k, l)$, the Joint Density of (B, C)

Theoretically, the most that could be hoped for is a complete recovery of the density $h(k, l)$ of the joint variable (B, C). We now investigate the extent to which this can be achieved. It is useful for us to first derive expressions for $c(k, l)$, which represents a finite horizontal bar (from $B = 0$ to $B = k$ at height l) in Figure 5.2. Using Equation (5.15), we can solve for $c(k, l)$ as follows:

$$c(k, l) = H(k, l) - H(k, l - 1)$$

We also define $F_r(s)$, the CDF corresponding to $f_r(\cdot)$, which is observable.

$$F_r(s) = \sum_{i \leq s} f_r(i) = P(S \leq s | R = r)$$

Geometrically, using Figure 5.2, $F_r(s)$ is composed of nested angle sets whose corners lie along a diagonal that goes down and to the left from $(s, s-r-x)$. Hence, $F_r(s)$ is the "rectangle set" $H(s, s-r-x)$. We can combine the above two equations to get

$$\begin{aligned} c(k,l) &= F_{k-l-x}(k) - F_{k-l-x+1}(k) \\ &= \sum_{i=0}^{k} [f_{k-l-x}(i) - f_{k-l-x+1}(i)] \qquad (5.20) \\ &= f_r(r+l+x) + F_r(r+l+x-1) \\ &\qquad - F_{r+1}(r+l+x), \qquad (5.21) \end{aligned}$$

where $r = k-l-x$.

To derive expressions for $h(k,l)$, we recall that $c(k,l)$ and $c(k-1,l)$ are both finite horizontal bars of different lengths at the same height. Hence, $h(k,l) = c(k,l) - c(k-1,l)$. Combining this relation with Equation 5.20 we can calculate the joint density as,

$$\begin{aligned} h(k,l) = &\sum_{i=0}^{k-1} [2f_{k-l-x}(i) - f_{k-l-x-1}(i) - f_{k-l-x+1}(i)] + \\ &[f_{k-l-x}(k) - f_{k-l-x+1}(k)] \qquad (5.22) \\ = &F_{k-l-x}(k) + F_{k-l-x}(k-1) - \\ &F_{k-l-x+1}(k) - F_{k-l-x-1}(k-1). \qquad (5.23) \end{aligned}$$

Equation (5.23) provides $h(k,l)$ using three values of r, namely $k-l-x$ and $k-l-x \pm 1$, and several values of s. The collection of (r,s) values required is illustrated in Figure 5.3(d).

Since $F_r(s)$ is undefined for $r < 0$, the above expression for $h(k,l)$ can be used only when $k-l-x \geq 1$. Hence, we in fact cannot determine individual $h(k,l)$ values for $k-l \leq x$! We call this region, marked with smaller rectangles in Figure 5.2, the *ambiguity zone*. However, Equation (5.20) shows that, for fixed l, we do know $c(l+x,l)$ which is the sum of $h(k,l)$ over $l \leq k \leq l+x$, that is the mass in an aggregate traversing the width of the ambiguity zone (an exception occurs when $l = -x$, where $h(0,l) = c(0,l)$ *is* known from Equation (5.20)). Note that other aggregates involving (k,l) values in the zone cannot be calculated. In particular, the marginal $b(k)$ of B cannot be determined.

The above can also be explained geometrically in terms of angle sets. The highest placed angles are those corresponding to $r = 0$, whose corners correspond to the diagonal comprising the lower

edge of the ambiguity zone. Since points in the interior of the zone, that is $l \geq k - x - 1$, cannot be corners, it follows that for a given $l \geq -x$ the *only* angles passing through points in the zone are those whose horizontal members align at $l = s - r - x$. Consequently, it is impossible to resolve the $h(k, l)$ values in the interior of the zone.

To understand *why* the corners cannot lie in the interior of the ambiguity zone, note that the horizontal component $c(s, s - r - x)$ of the angle set (r, s) (refer to Equation (5.15)) corresponds to (k, l) pairs such that the two probes share the same busy period, whereas the vertical component $b(s, s - r - x - 1)$ contains scenarios where they do not. Because of the invasive impact of the probe size x however, they **must** be in the same busy period if $l \geq k - x$, which is precisely the definition of the ambiguity zone.

We conclude with a comment on the role of x. Since increasing x widens the ambiguity zone and, if we use an intrusive probing stream, increases delays *without impacting* the density $h(k, l)$, it serves to increase the available range of r values, thereby improving the applicability of expressions in Class 1. However, smaller x reduces the ambiguity zone and enables h to be determined more fully.

5.4 Approximate Inversion Expressions: Class 2

The Class 1 expressions for $c(l)$ above relied on $r \geq t - x$. Thinking ahead to practical situations with limited data, such values may be rare or even entirely unavailable. Hence, we develop inversion expressions that do not require the above lower bound on r. However, they are approximate and to distinguish them from the exact Class 1 expressions, we call them Class 2 expressions. They are approximate because they assume that certain portions of the strip have zero density. In particular, we assume that $h(k, l)$ is sufficiently concentrated at small k and l. Our Class 2 expressions use two kinds of approximations - weak and strong. They differ from each other in the portion of the strip that, they assume, has zero density. In this section, we define each of these assumptions and derive approximate inversion expressions based on them.

5.4.1 Weak Assumption

Consider Equation (5.20) for a given l. If in fact $h(k,l)$ is negligible for $k > k_l$, where $k_l \in [l, l+t-1]$ lies inside the strip, then $c(l) = c(k,l)$ whenever $k \geq k_l$. We refer to this as the *weak assumption*. It is reasonable to expect that it holds in many cases, at least for sufficiently large l, as increasing k corresponds in some sense to "tail" events of low probability. It leads to the following approximation for each fixed r obeying $r \geq r_l = k_l - l - x$:

$$c(l) \approx F_r(r+l+x) - F_{r+1}(r+l+x), \quad \text{assuming } h(l+r'+x, l) = 0 \quad \forall\, r' > r. \tag{5.24}$$

Equation (5.24) uses only a single $r \geq r_l$. As in the previous section, it makes sense to combine different values to reduce variability. Using uniform weights results in a satisfying cancellation of $f_r(s)$ terms, which would not occur if weighted averaging were used. Using r now as a parameter obeying $r \geq r_l$, we obtain:

$$\begin{aligned} c(l) &= \frac{1}{N} \sum_{r'=r}^{r+N-1} [F_{r'}(r'+l+x) - F_{r'+1}(r'+l+x)]. \\ &= \frac{1}{N} \sum_{r'=r}^{r+N-1} f_{r'}(r'+l+x) + \\ &\quad \frac{F_r(r+l+x-1) - F_{r+N}(r+N+l+x-1)}{N}. \end{aligned} \tag{5.25}$$

The first term is the average of N expressions of the form of Equation (5.17), whereas the second includes $h(k,l)$ values that are not affected by the weak assumption. In Figure 5.3(b) the (r,s) values required by the extra terms appear as the vertical lines.

5.4.2 Strong Assumption

If k_l is such that $h(k_l, l)$ can be considered a tail probability, it is reasonable to expect that this may also be true of $h(k_l, l')$ for l' values less than l. This motivates the following *strong assumption*, which in addition to the weak assumption, supposes that $h(k', l')$ vanishes for all $l' < l$ when $k' \geq k_l$, or equivalently $r' \geq r_l$. In other words, all elements directly below, and below and to the right of, the point (k_l, l). It is not difficult to see that, for a fixed r obeying $r \geq r_l = k_l - l - x$,

this leads to

$$c(l) \approx f_r(r+l+x), \tag{5.26}$$
$$\text{assuming } h(l+r'+x,l)=0 \quad \forall r' > r$$
$$\text{and } h(l'+r'+x,l')=0 \quad \forall r' \geq r,\ l' < l.$$

As we did with Equation (5.24), we can average N consecutive values starting from a given $r \geq r_l$, yielding

$$c(l) \approx \frac{1}{N} \sum_{r'=r}^{r+N-1} f_{r'}(r'+l+x). \tag{5.27}$$

The leftmost angle in Figure 5.2 is an example of the terms (angles) in this sum whose corner lies inside the strip. The stronger assumption has resulted in the loss of the difference term of Equation (5.25). The corresponding plot Figure 5.3(c) shows that the vertical lines have vanished, leaving a diagonal set similar to Figure 5.3(a), only with uniform weights.

We can also perform averaging with a natural set of weights as we did with the class 1 inversion methods. Recall that $g_r(u) = P(S-R=u|R \geq r)$, and let $p_r(r')$ denote the conditional probabilities $P(R=r'|R \geq r)$, interpreted as a set of weights which sum to 1. Using the strong assumption one can show that

$$\begin{aligned} c(l) &\approx \sum_{r'=r}^{\infty} f_{r'}(r'+l+x)p_r(r') \\ &= \sum_{r'=r}^{\infty} P(S=l+x+R|R=r')P(R=r'|R \geq r) \\ &= \sum_{r'=r}^{\infty} P(S-R=l+x|R \geq r) \\ &= g_r(l+x). \end{aligned} \tag{5.28}$$

This is formally the same expression as Equation (5.19)! However, r is smaller than $t-x$, and the corners of the angles set involved are inside the strip. Just as we did for $f_r(s)$, we can define a CDF

corresponding to $g_r(u)$, and establish the following identity:

$$
\begin{aligned}
G_r(u) = & \sum_{i=x-t}^{u} g_r(i) = P(U \leq u | R \geq r), && (5.29)\\
= & \sum_{l'=-t}^{l} g_r(l'+x), \\
= & \sum_{l'=-t}^{l} \sum_{r'=r}^{\infty} f_{r'}(r'+l'+x) p_r(r'), \\
= & \sum_{r'=r}^{\infty} p_r(r') \sum_{l'=-t}^{l} f_{r'}(r'+l'+x), \\
= & \sum_{r'=r}^{\infty} p_r(r') F_{r'}(r'+l'+x), && (5.30)\\
= & \sum_{r'=r}^{\infty} p_r(r') H(r'+l+x, l), && (5.31)
\end{aligned}
$$

which shows that the CDF corresponding to $g_r(u)$ can be viewed as a weighted sum of rectangle sets, with the same (l independent) weights as before.

As with class 1, increasing x has the advantage that a fixed (r, s) observation now corresponds to an angle which is lower in the strip, whereas the density h, being independent of probes, has not changed. Hence, this same (r, s) pair is now more likely to fall into a region where the weak and strong assumptions hold than before. Each of these advantages is consistent with the intuition that larger probe sizes increase the chances of probes being in the same busy period, where information can be extracted about $c(l)$. Of course, the disadvantage is the widening of the ambiguity zone which decreases the observability of h by increasing the width of the ambiguity zone.

5.5 Multi-Hop Extensions

So far, we rigorously analyzed and developed inversion expressions to access properties of cross-traffic flowing across a single FIFO hop. In Section 5.2.3, we discussed that sample path ambiguity can result at all but very small intra-pair separations. But, under a well-motivated assumption, almost complete inversion of the two cross-traffic functionals is possible. It is natural to ask if such if such inversion is possible with multi-hop paths. We investigate this by considering the simplest multi-hop path - a path with two hops. We find that, even with such a two-hop path,

inversion is complicated by two factors - cross-traffic persistence and too many "unobservables". We revisit these factors by investigating how they can be eliminated, in Chapter 7.

Using the convention of the previous chapter, let $T'_{n,i}$ denote the departure time of the i^{th} packet of the the n^{th} pair from the first hop. For simplicity, we assume no propagation delay and hence, this is also the arrival time at the second hop. Let $T''_{n,i}$ be the departure time of the same packet from the second hop. Also, let x_i denote the transmission time of both probe packets at the i^{th} hop. A_i denotes the measure representing cross-traffic arrival at hop i. Then, we can write the following by applying Equation 5.1 to the second hop.

$$T''_{n,2} = x_2 + \big[\big(T''_{n,1} + A_2([T'_{n,1}, T'_{n,2}))\big) \vee \sup_{v \in [T'_{n,1}, T'_{n,2}]} \big(v + A_2([v, T'_{n,2})\big)\big] \tag{5.32}$$

Subtracting $T_{n,2}$ from both sides, we get the following expression relating the delay $D_{n,2}$ of the second probe with the delay $D_{n,1}$ of the first probe.

$$D_{n,2} = x_2 + \big[\big(D_{n,1} + A_2([T'_{n,1}, T'_{n,2})) - (T_{n,2} - T_{n,1})\big)\big) \vee \sup_{v \in [T'_{n,1}, T'_{n,2}]} \big(A_2([v, T'_{n,2})) - (T_{n,2} - v)\big)\big]$$

Let Δ'_n denote $T'_{n,2} - T'_{n,1}$, the difference in the departures from the first-hop of the packet pair, we get

$$\begin{aligned} D_{n,2} = x_2 + \big[\big(D_{n,1} + A_2([T'_{n,1}, T'_{n,1} + \Delta'_n)) - t\big)\big) \vee \\ \sup_{v \in [T'_{n,1}, T'_{n,1} + \Delta'_n]} \big(A_2([v, T'_{n,1} + \Delta'_n)) - (t + T'_{n,1} - v) + (T'_{n,1} - T_{n,1})\big)\big] \end{aligned} \tag{5.33}$$

5.5.1 Cross-traffic Persistence

In Equation 5.33, the time interval over which the cross-traffic of the second hop appears begins with $T'_{n,1}$ which is dependent on $A_1(.)$, the cross-traffic arrival process at the first hop. If cross-traffic was only 1-hop persistent, i.e., traversed only one hop of the path, A_1 and A_2 are independent. In such a case, stationarity would imply that the cross-traffic arrival from $T'_{n,1}$ onwards would statistically be the same as the cross-traffic arrival from time 0. Hence, Equation 5.33 can be written in the 1-hop persistent case as,

$$\begin{aligned} D_{n,2} = x_2 + \big[\big(D_{n,1} + A_2([0, \Delta'_n)) - t\big)\big) \vee \\ \sup_{v \in [0, \Delta'_n]} \big(A_2([v, \Delta'_n)) - (t - v)\big) + (T'_{n,1} - T_{n,1})\big] \end{aligned} \tag{5.34}$$

However, if all cross-traffic to the first hop persists to the second hop, then $A_2([T'_{n,1}, T'_{n,1} + \Delta'_n))$ is equal to $A_1([T_{n,1}, T_{n,2}))$ scaled by λ, the ratio of the capacities of the two hops. The scaling factor is required because we measure cross-traffic in units of transmission time. Therefore, in this persistent cross-traffic case, Equation 5.33 becomes,

$$D_{n,2} = x_2 + \big[(D_{n,1} + \lambda A_1([0,t)) - t)) \vee \sup_{v \in [0,\Delta'_n]} (\lambda A_1([f(v),t)) - (t-v)) + (T'_{n,1} - T_{n,1})\big] \quad (5.35)$$

where, $f(v)$ is a function such that $A_1([f(v),t))$ always counts the same cross-traffic as $A_2([v,\Delta'_n))$ and vice versa.

The exact expressions of Equation 5.34 and 5.35 are less relevant to us than the effect persistence has on the system evolution. In general, part of $A_1(.)$ may persist to $A_2(.)$ in which case the system is described by an unknown combination of both these expressions.

5.5.2 Unobservable Δ'_n and $T'_{n,1}$

Cross-traffic persistence is not the only reason why the two-hop case is hard to analyze. To see why, assume that we know how persistent cross-traffic is. Say, for instance, it is entirely persistent and Equation 5.35 is true. Even if λ and $f(.)$ are known, there are two unobservable quantities, $T'_{n,1}$ and Δ'_n. These represent information about the probe arrivals to the first hop. In practice, a very rough approximation of these quantities would be the times at which ICMP replies to TTL-expired "bounding" packets are received. This is known to be inaccurate due to the various known issues with ICMP replies [LB00, PV02a]. Moreover, dependence between these unobservables and the observable $D_{n,i}$ makes it unlikely that easy-to-justify assumptions, e.g., our independence assumption, can be used either.

Above, we motivated why it is inherently challenging to design techniques that are non-intrusive *and* applicable for use with true multi-hop paths, i.e., paths that cannot be approximated as a single-hop path. We did not present a formal proof. But, by laying out the reasons why a multi-hop analysis is difficult, we provide guidance to future work that attempts to solve similar problems. Indeed, this discussion is a starting point for our design of a measurement-friendly network architecture in Chapter 7.

5.6 Conclusions

In this chapter, we tackled the question of the in-principle potential of packet pair methods in identifying statistics of cross-traffic. First, we analyzed the 1-hop FIFO case and showed using simple queueing theory that probe pair delays expose two functionals of cross-traffic. These functionals can be viewed as representing the average amount and burstiness of cross-traffic. Then, we showed that, the system is sample path ambiguous unless the intra-pair separation is very small. To do away with the latter requirement and not use specific parametric models of cross-traffic, we used the assumption that the delay of the first probe is independent of the cross-traffic functionals between it and the second probe of the pair. Under this well-motivated assumption, we showed that almost complete inversion, of the two cross-traffic functionals is possible. We derived inversion expressions to access the average cross-traffic and the joint density of the two functionals. We derived exact Class 1 and approximate Class 2 inversion expressions. Finally, we showed that cross-traffic persistence and the unobservability of probe traversal at various hops makes a similar analysis of the two-hop case fundamentally hard.

In the next chapter, we design practical estimators based on the inversion expressions derived in this chapter. We find the Class 2 expressions much more useful, in practice, than the Class 1 expressions. We use a combination of simulations, traces and live experiments to evaluate our estimators as well as justify our assumption. Our illustration of the hardness of analyzing multi-hop paths guides us in the design of measurement-friendly networks, in Chapter 7.

Chapter 6

Practical Inversion: Cross-traffic Estimation

"In theory, there is no difference between theory and practice; In practice, there is."

- "Yogi" Berra, American baseball player and manager

In the previous chapter, we derived inversion expressions to estimate statistics of two cross-traffic functionals in the single-hop case. In this chapter, we use these expressions to define and evaluate practical estimators of cross-traffic. Our primary focus is on estimating the distribution of the functional representing the amount of cross-traffic in an intra-pair time interval. In designing estimators for this purpose, we encounter various challenges including those related to data availability and monotonicity of estimated CDFs. We also provide illustrative results showing that the joint distribution of the two cross-functionals can be estimated. We use a combination of simulations, traces and live experiments to refine and evaluate our estimators. Our results show that not only can we estimate the entire distribution of cross-traffic but we can also do so using intra-pair gaps that are much larger than those used by prior methods [SKK03]. For instance, on links that have a utilization as low as 50%, we can use an intra-pair gap that is up to ten times the transmission time of the largest packet.

This chapter is organized as follows. In Section 6.1, we use key inversion expressions to formally define the estimators we use. All estimators use a parameter $\mathbf{r}$ to select only some of the

probe pairs. A major portion of this chapter is devoted to investigating how **r** may be chosen. In Section 6.2, we use simple *qsim* simulations to understand when and how the data available, to estimate the density of the amount of cross-traffic, varies. We find a classic bias-variance trade-off with our estimators that also illustrates the advantage of estimators based on Class 2 inversion over estimators based on Class 1 inversion. In Section 6.3, we move to estimating the CDF (as opposed to density) and show the advantages of adapting **r** for estimation. We also show the advantage of the strong assumption over the weak assumption in estimating a naturally-normalized CDF and present a modified definition of the strong assumption that is more appropriate for estimating CDFs. In Section 6.4, we use realistic simulations to develop and compare different varieties of practical estimators with an adaptive **r**. We investigate a saturation algorithm to estimate the strong assumption curve and use this curve to choose **r**. Since this does not output a monotonic CDF, we investigate various monotonization algorithms. We also develop a simpler algorithm that automatically estimates monotonic CDFs by using (adaptively-chosen) constant **r**. We evaluate the various estimators that we develop in Section 6.5. We find that our adaptive estimator performs well though the monotonicity variants do not have much impact on performance. In Section 6.6, we present the results of evaluating the various estimators using packet traces and novel live experiments. We find that, for utilization as low as 0.5, we can estimate CDFs with a (root mean square) error less than 0.2 for large intra-pair separations. In Section 6.7, we discuss important practical details such as clock synchronization. We also show describe natural ways of making our estimators (and prior works) more accurate by trading off non-intrusiveness. We summarize our contributions in Section 6.8, namely, how large intra-pair separations can be used to accurately estimate cross-traffic in practice.

6.1 From Inversion to Estimators

In this section, we use the inversion expressions derived in the previous chapter to develop practical estimators of cross-traffic. We develop three estimators of the density, $c(l)$, of C, the cross-traffic functional representing the *net* amount of cross-traffic in an intra-pair time interval. We also develop an estimator of $h(k, l)$, the joint distribution of B and C, the two cross-traffic functionals that relate the delays of a pair of probe packets.

We construct the initial estimators in two steps. First, the observable conditional densities $f_r(s) = P(S = s|R = r)$ and $g_r(u) = P(S - R = u|R \geq r)$ are estimated. To do so, we simply use the empirical frequencies, denoted by $\hat{f}_r(s)$ and $\hat{g}_r(u)$ respectively, based directly on the observed packet pair delays (R, S). If there are no samples for a given $r = r'$, which is often the case even though $f_r(s)$ is typically positive, we set $\hat{f}_{r'}(s) = 0$ for all s except the largest where we set the density to 1. If there are none for all $r \geq r'$, then $\hat{g}_{r'}(u) = 0$ for all u, except the largest u. Such estimators are intuitive, and enjoy the property that they are naturally normalized. By this we mean that their empirical CDFs, $\hat{F}_r(s) = \sum_{i=0}^{s} \hat{f}_r(i)$, and $\hat{G}_r(u) = \sum_{i=x-t}^{u} \hat{g}_r(i)$, monotonically increase from 0 up to 1, as a CDF should. In the second step, we select inversion expressions from the previous section, and replace each of the exact observables $f_r(s)$, $g_r(u)$ and $F_r(s)$ by their estimated counterparts.

The first estimators of $c(l)$ we consider are defined below in Equations (6.1) through (6.3). The symbol **r** is used for the free parameter rather than r, to avoid confusion with the latter's use as a sample of the excess delay variable R. Recall that since l is fixed when estimating $c(l)$, specifying **r** is equivalent to setting $\mathbf{k} = \mathbf{r} + l + x$, defining the corner $(k^*, l^*) = (\mathbf{k}, l)$ of the leftmost angle used by the estimator.

$$\hat{c}_1(l) = \hat{g}_{\mathbf{r}}(l + x). \tag{6.1}$$

$$\hat{c}_2(l) = \frac{1}{N} \sum_{r'=\mathbf{r}}^{\mathbf{r}+N-1} \hat{f}_{r'}(r' + l + x). \tag{6.2}$$

$$\hat{c}_3(l) = \frac{1}{N} \sum_{r'=\mathbf{r}}^{\mathbf{r}+N-1} \hat{f}_{r'}(r' + l + x) + \frac{\hat{F}_{\mathbf{r}}(\mathbf{r} + l + x - 1) - \hat{F}_{\mathbf{r}+N}(\mathbf{r} + N + l + x - 1)}{N}. \tag{6.3}$$

Estimator $\hat{c}_1$ arises from Equation (5.19). It differs from that equation however in that **r** is not set to $t - x$, corresponding to the largest range of r under Class 1, but is free to take any value both below and above $t - x$. It can therefore be seen to be of either Class 1 or 2, depending on the situation. We have encountered its Class 2 form already in Equation (5.28).

Estimator $\hat{c}_2$ arises from Equation (5.27). Again, we do not prescribe the value of the parameter **r**, but allow it to "operate" as either Class 1 or Class 2. (see Equation 5.18 and Equation 5.27 respectively). $\hat{c}_2$ can be naturally contrasted to estimator $\hat{c}_1$ using the same **r** value.

Estimator $\hat{c}_3$ arises from Equation (5.25) which we derived using the weak assumption. It is similar to $\hat{c}_2$ but with the addition of correction terms. Unfortunately, these terms have a negative impact on estimation performance, as we show later.

Estimators 2 and 3 use $r = \mathbf{r}$ to $r = \mathbf{r} + N$. To ensure that, for a given $\mathbf{r}$, they use the same amount of information as $\hat{c}_1$ we take N large enough so that each uses all observations of $R \geq \mathbf{r}$. This simplifies comparison and eliminates the need to perform parameter selection with respect to N.

We now define our estimator for $h(k,l)$. We must distinguish between points in the ambiguity zone and those that are not. Outside the zone, that is for $k - l - x \geq 1$, the estimator arises from Equation (5.23) and is

$$\hat{h}(k,l) = \hat{F}_{k-l-x}(k) + \hat{F}_{k-l-x}(k-1) - \hat{F}_{k-l-x+1}(k) - \hat{F}_{k-l-x-1}(k-1). \tag{6.4}$$

The estimator for the horizontal aggregates in the ambiguity zone follows from Equation (5.20):

$$\hat{c}(l+x,l) = \hat{F}_0(l+x) - \hat{F}_1(l+x). \tag{6.5}$$

6.2 Fundamental Properties

Figure 6.1 shows the steps that we take in the next three sections to refine our estimators. In this section, we investigate the fundamental properties of our estimators assuming that our assumed model is perfect. To achieve this, we use simulations with simple arrival processes, using a minimum number of parameters, to illustrate important factors affecting the estimators above. We use more realistic workloads in later sections of this chapter.

This section is organized as follows. In Section 6.2.1, we describe the simple simulation setup that we use. In Section 6.2.2, we use illustrative simulations to show how the observed data changes with respect to cross-traffic intensity (or equivalently, link utilization) and burstiness. These help explain the performance of our estimators later in the chapter. In Section 6.2.3, we show that our $c(l)$ estimators possess a classic bias-variance trade-off, i.e., any attempt to reduce variance by using

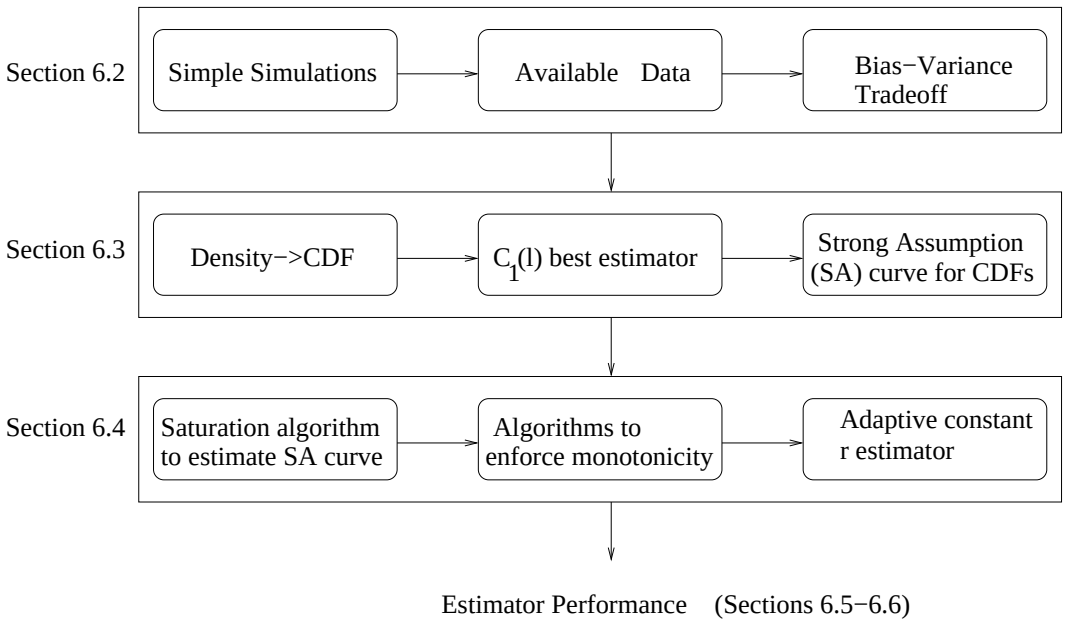

Figure 6.1. Flowchart illustrating the steps we take in developing our final estimators. In Section 6.2, we use simple simulations with constant packet sizes to illustrate how available data changes with cross-traffic intensity and burstiness. We also illustrate the bias-variance trade-off. In Section 6.3, we shift from density estimation to CDFs and define the strong assumption more appropriate for CDFs. Moreover, among our three estimators, the CDF corresponding to $c_1(l)$ is the best. We also motivate how adaptivity in choosing **r** can have significant advantages. In Section 6.4, we describe a saturation algorithm to estimate the strong assumption curve. Then, we use this algorithm to propose a composite estimator that requires additional algorithms to ensure the monotonicity of CDFs. We also describe a simpler adaptive estimator that does not require monotonicity. We evaluate all of these estimators in the later parts of this chapter.

more data inevitably increases bias and vice versa. This trade-off illustrates the innovation of our approach. In the past, delay observations were divided into two categories according to an (inferred) busy period/idle period criterion for probes [PV02b]. Although this is a very intuitive approach well suited to heuristic methods, it is difficult to carry it further. Instead, we employ conditioning with respect to R, thereby providing a sequence of subsets of probe delay, indexed by r. These subsets vary in their proportions of probes pairs which share, or not, the same busy period. From each subset an estimate can be obtained. For large enough r, the sets fall into Class 1 and therefore contain only the busy cases, so that estimates based on these are bias free. As we enter Class 2 at smaller r, this is no longer the case but it remains approximately so until the strong assumption is broken. By viewing the problem in this way, we have a sequence of estimators of steadily decreasing bias but

increasing variance as r increases. Thus, there are significant benefits of choosing **r** adaptively - a question we address in the next two sections.

6.2.1 Simulation Setup

As with the analysis of Section 5.3, we simulate a fully discretized single-hop system, with slotted time corresponding to the transmission time $\delta = 8d/\mu$ [sec], where $d = 10$ is the size of the slot measured in bytes, and μ is the output link capacity. We use the *qsim* simulator described in Chapter 3. Cross traffic arrivals are taken to be Poisson, with constant packet size p bytes. In our discrete setting, the number of Poisson arrivals in each slot is an i.i.d. Poisson random variable with parameter $\lambda\delta$. The packet transmission times are also integer multiples of δ. The above cross-traffic and packet size combination corresponds to a particularly simple example of a stationary measure A that satisfies the assumption of (B, C) being independent of R due to the "memoryless" nature of Poisson streams.

We use periodic probing streams with probes of size $p = 40$ bytes too, so $x = p/d = 4$, with period $t = 10p/d = 40$ slot units, or the time taken to transmit 400 bytes. Due to the Poisson nature of cross-traffic, periodic probing streams achieve joint ergodicity [MVBB05, MVB$^+$05]. In this section and the next, it will be convenient to present results either as integers from the slotted discrete time system, l, k, r, etc. (already normalized by δ), or in units of bytes.

6.2.2 The Issue of Available Data

To understand how the estimators behave, it is essential to know the environment they operate in. The following paragraphs examine this in detail for the system described above with cross-traffic intensity, $\rho = 0.8$ and $\delta = 0.25$ [ms] corresponding to $\mu = 320$Kbps.

We begin with Figure 6.2, where the shaded area visualizes the joint density $h(k, l)$ of (B, C). The density is concentrated on lines corresponding to whole numbers of packets, and lies away from the lower edge of the strip for almost all l values. This indicates that the weak and strong assumption will hold for many corners (k^*, l^*) well inside the strip. The density is particularly concentrated near the l axis, corresponding to low per-slot "burstiness" of the cross-traffic. The

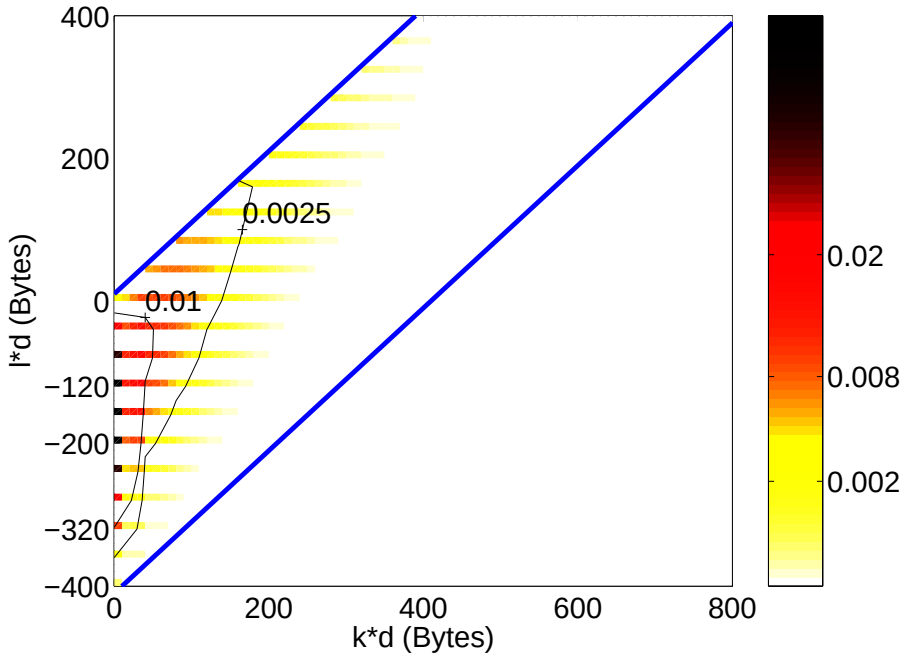

Figure 6.2. The density $h(k,l)$ of (B,C) for $\rho = 0.8$ (darker tones indicate higher density), and corresponding contour lines giving probability per 'pixel'. The density is concentrated on discrete l values corresponding to whole numbers of packets. The two values of l used in Figure 6.6, $-120/d$ and $-320/d$, are shown (d is the number of bytes per time slot).

superimposed contour lines give an idea of the probabilities corresponding to the "pixels" of this shading, which were drawn at full slot resolution. [1]

Knowing where the density becomes negligible tell us which r values are necessary to measure it, and therefore informs the choice of the parameter value **r** which controls the range of r used by the estimators. The next question is, how available are these desirable **r** values? Or equivalently, what is the probability that the angle sets they correspond to will be seen? It is instructive to first visualize the density $m(r,s)$ of the packet pair delays (R,S), as seen in the left plot in Figure 6.3. Mass is concentrated on lines of constant $u = r - s$ at large r, since there the queue cannot empty between the consecutive probes corresponding to r and s and so they must share a busy period. The probe separation u is then constrained to be multiples of a cross-traffic packet transmission time.

To see how the density $m(r,s)$ impacts estimation, it is more useful to transform this information on "available data" into a form which is directly readable in the (k,l) plane. Recall that there

[1] For visual clarity, the contour lines, here and elsewhere, were smoothed to emphasize the l values corresponding to whole numbers of packets. At other l values the contours cut in much closer to the l axis.

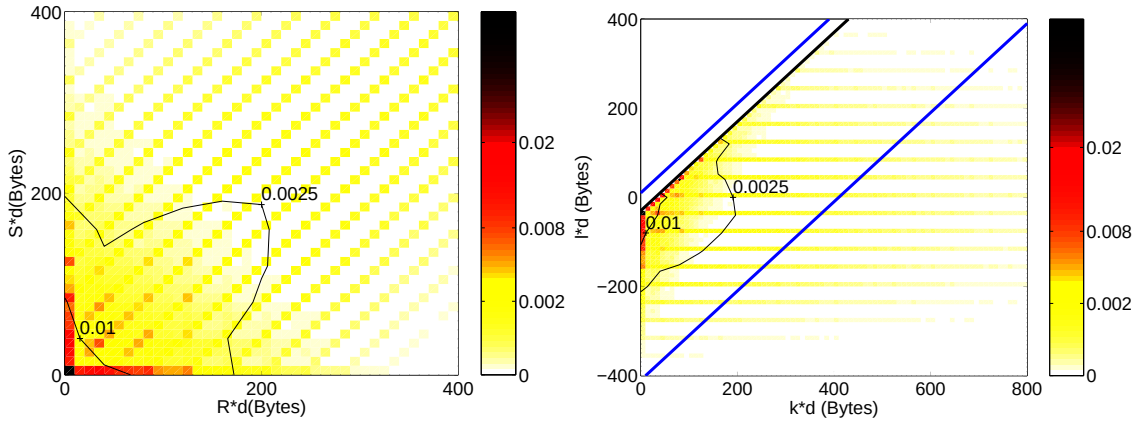

Figure 6.3. The density $m(r, s)$ of (R, S) (left), and its transformation into *angle density* $a(k, l)$ (right). Darker tones indicate higher density. Contour lines are for probability per "pixel". Lines of constant $u = r - s$ are mapped to horizontal lines in the (k, l) plane. The ambiguity zone is between the the top of the strip and the diagonal line immediately below it. There is no angle density in the ambiguity zone.

is a 1-1 mapping between (r, s) pairs and angle corners: $(k^*, l^*) = (s, s - r - x)$. Applying this mapping to $m(r, s)$ induces what we call the *angle density*, $a(k, l)$. The right plot of Figure 6.3 displays the angle density, together with corresponding contour lines. It allows us to directly see the available data for each angle set. The affine mapping has taken vertical/diagonal/horizontal lines in (r, s) space and mapped them to diagonal/horizontal/vertical lines in (k, l) space respectively.

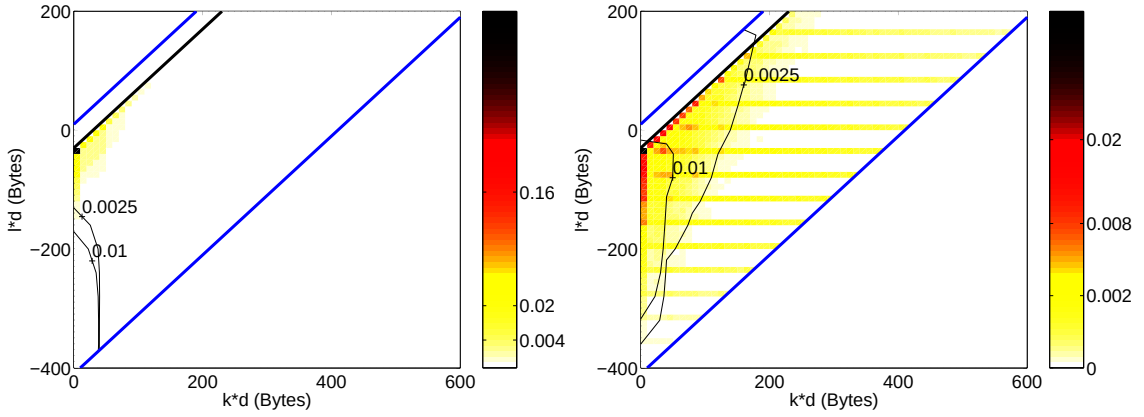

Figure 6.4. Superimposing angle density onto contours of $h(k, l)$ for $\rho = 0.2$ (left) and $\rho = 0.8$ (right). The shading of the right plot is a zoomed-in version of the shading in the right plot of Figure 6.3. The angle density is given by the shading, and the cross-traffic density by two contour lines. The degree of coverage of h by the angle density varies significantly.

Figure 6.4 gives a representation of angle density $a(k, l)$ and the cross-traffic density $h(k, l)$ in

the same plot, for a low (0.2) and high value of ρ. To avoid overcrowding the figure, the shading is given for angle density with no contours, and two contours are given for h, with no shading. As is clearly seen in the low ρ case, where h is concentrated is not necessarily where the angle density is located. More generally, it is clear that to resolve h well across the (k, l) plane, we need a sufficient "coverage" of angle density, and that whether this is achieved will depend on the number of observations as well as the queueing statistics (which are a function of ρ, or more precisely, of the combined cross-traffic and probe traffic processes).

Angle density shows where the "available data" is located in the strip, however estimators make use of *sets* of angles. Thus, to see what is actually available for an estimator to use, we must sum over these sets. Recall from the previous chapter (and Equation 5.18) that each choice of l and $\mathbf{r}$ designates a set of angles whose corners lie at (k, l) where $k \geq \mathbf{k} = \mathbf{r} + l + x$. The mass contained in these angles, that available to an estimator, is simply the sum of the angle density $a(k, l)$ contained in the horizontal segment defined by a given fixed l and $k \geq \mathbf{k}$. This is also equivalent the sum of densities $m(r, s)$ such that $s - r = \mathbf{u} = l + x$ and $r \geq \mathbf{r}$. The right plot in Figure 6.5 repeats

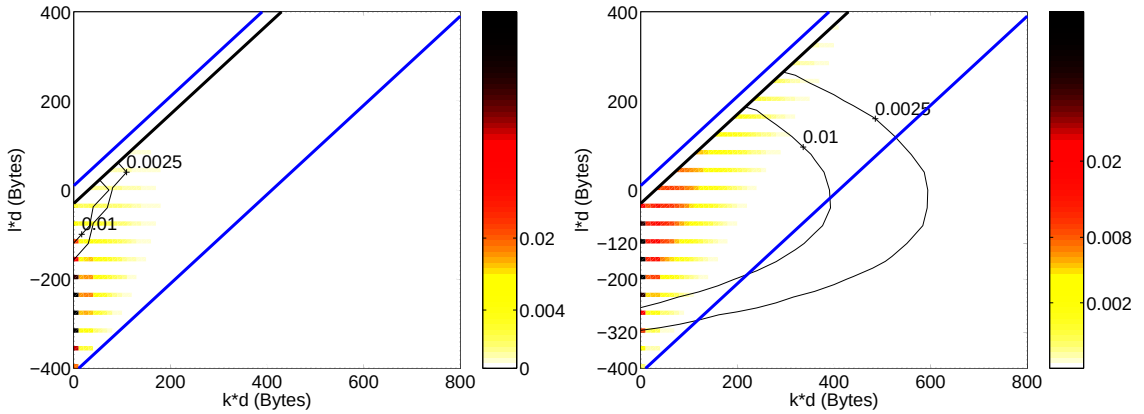

Figure 6.5. Superimposing contours of the 'available mass' used by estimators, over the density $h(k, l)$, for $\rho = 0.4$ (left) and $\rho = 0.8$ (right). We use the same contours on both plots to contrast the two ρ values.

the $h(k, l)$ density from Figure 6.2 without the contours. Instead, contours are drawn based on the density of *available mass* used by an estimator as just defined. Using them, for any given l we can easily see which regions in the strip are data rich or data poor from the point of view of an estimator of $c(l)$. The answer clearly depends on l. The left plot in Figure 6.5 shows a similar plot for $\rho = 0.4$, where the coverage is worse.

In conclusion, we are left with two important questions: whether strong or weak assumptions hold (and the shape of these regions), and whether there is sufficient data available to an estimator, depend strongly on l. Furthermore, the interplay between the densities $h(k,l)$ and $a(k,l)$ over the plane is crucial. The feasibility of estimation depends strongly on l, and furthermore there is an intrinsic difficulty in that the degree of coverage may not be adequate. For low levels of burstiness/utilization, the marginal of R is concentrated near $R = 0$, resulting in available mass which is strongly concentrated near $(k,l) = (0,-x)$. But, under these same circumstances, $h(k,l)$ is concentrated near $(k,l) = (0,-t)$. The overlap of the two is small, and any estimator will have great difficulty, essentially because the region where the data is needed in order to measure the (B,C) values which occur, is precisely where data is scarce. Coverage is determined not by utilization alone but by the spread of (r,s) values seen, which depends on ρ, burstiness, as well as the number of observations.

6.2.3 A Bias/Variance Trade-off at Fixed l

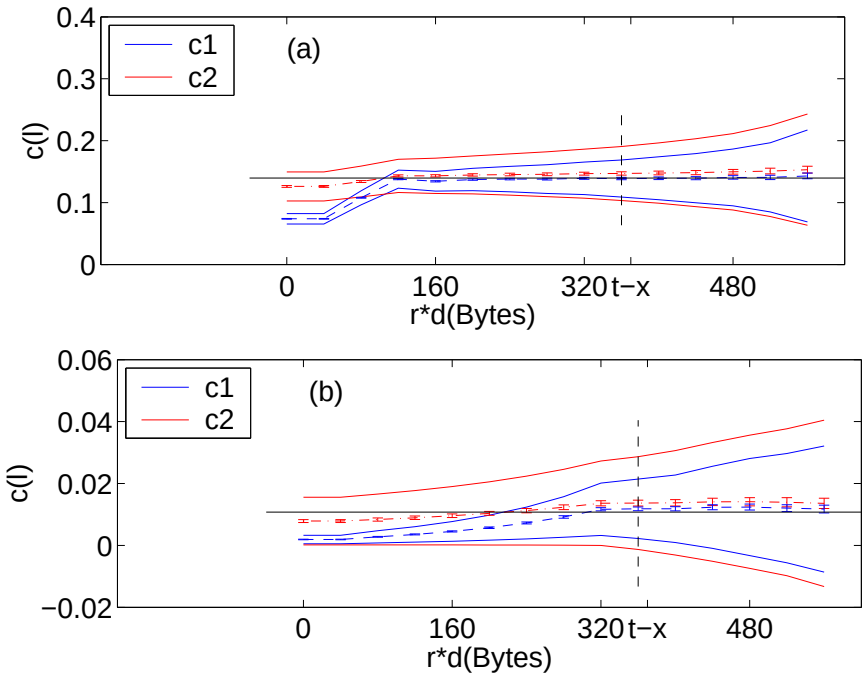

Figure 6.6. Estimator bias and standard deviation as a function of **r**. The horizontal line is the true value of $c(l)$. (a) $l = -3p/d$, or -120 bytes. The bias begins at -80 bytes. (b) $l = -8p/d$, or -320 bytes. The bias begins at -280 bytes.

We now examine estimator performance for $l = -3p/d$, or -120 bytes in the $\rho = 0.8$ case. For this value of l, the right plot in Figure 6.5 shows that conditions are good: the available mass contour lines shows that the region where $h(k,l)$ is concentrated is well covered, and also that there is considerable mass available in the angles both inside and to the right of the strip. Furthermore, $h(k,l)$ is small for a considerable distance to the left of the bottom edge of the strip. Hence both $\hat{c}_1$ and $\hat{c}_2$ can be expected to perform well either as Class 1 or Class 2. Figure 6.6(a) compares the mean and standard deviation of the estimators, based on $n = 1000$ probes, as a function of the parameter **r**. The mean and standard deviation were estimated using $N = 1000$ independent experiments, each yielding a single sample for each estimator (and for each **r**). The estimated mean is shown with 1.96σ confidence intervals near the center of the plots. The outer curves are drawn one standard deviation (of the estimator) to either side of the means.

For $r \geq t - x$ each estimator operates as Class 1. As expected each gives approximately unbiased estimates of $c(l)$ in this case. As expected, the uniform weighting scheme of $\hat{c}_2$ is less effective, resulting in larger variance. The correction terms in Equation (6.3) separating $\hat{c}_2$ and $\hat{c}_3$ identically cancel under Class 1 in theory, however estimates of them do not. Effectively an imperfect estimate of zero is added, resulting in increased bias and variance (not shown).

For $r < t - x$ each estimator operates as Class 2. As **r** decreases, we expect each to become biased. This is indeed what is observed. However, there is **no sharp change** at $r = t - x$, since the strong assumption holds very well. For example a bound on the total error due to the assumption: mass ignored plus the undesirable mass included in the angle at $(\mathbf{k}, l)$, is only 0.1% of $c(l)$ (i.e. $(c(l) - c(\mathbf{k}, l) + b(\mathbf{k}, l-1))/c(l) = 0.001$). At small **r** however when the strong assumption finally fails, the bias of $\hat{c}_1$ is much worse, because its weighting scheme was not designed to cope with the errors inherent in Class 2. As before, the variance of $\hat{c}_1$ is lower, as greater weight lies in the data rich zones where **r** and s are smaller. Again, the correction terms of $\hat{c}_3$ worsen its performance (not shown) relative to $\hat{c}_2$. Since they are in the form of a difference of two quantities of similar size, they are sensitive to errors.

In conclusion, there is a classic bias variance trade off operating, which begins once the strong assumption ceases to hold. This value of **r**, which plays the role of the *effective* Class 1/Class 2 boundary, is clearly visible in Figure 6.6(a) as the point where the bias of $\hat{c}_1$ begins to be noticeable.

To the left of this point, c_1 has rapidly decreasing variance at the cost of increased bias, whereas $\hat{c}_2$ reacts more slowly. Figure 6.6(b) shows an analogous study for a smaller $l = -8p/d$, or -320 bytes. Similar conclusions on bias and variance hold for each estimator separately. But, the point marking the beginning of the bias increase now occurs earlier at $r = -8p/d$, again in line with the failure of the strong assumption as seen visually in the right plot of Figure 6.5.

Consider now the effect of reducing n, the number of probes, on the results in Figure 6.6. To first order, the qualitative behavior remains the same, but the standard deviation of estimates increase. To mitigate this, one could be led to select smaller values of $\mathbf{r}$ to capture more data, and to control the resulting increase in bias, c_2 may becomes more attractive in comparison to c_1. Thus, which estimator is preferable (at least in terms of bias), and in particular the success of a Class 1 inspired approach, is also dependent on the global amount of data available.

The above demonstrates the value of the Class 2 inspired estimators, and the necessity on not insisting on *always* having linear behavior of the queue for estimation. In earlier packet pair methods it was crucial that the probes share the same busy period **and** were back to back. Here this is not the case. Class 2 based estimators not only can be used, but they may even out-perform those based on Class 1 ideas. Furthermore, because of this, greater values of t can be used than before, because the push into the data-poor regime, to the detriment of Class 1 estimators, is no longer a fundamental barrier.

6.3 Motivating Adaptation

In this section, we move from the largely illustrative setting that we have used so far to more realistic and practical settings (see Figure 6.1). First, we begin to use estimators of the CDF and not density. We do this mainly because individual $c(l)$ values may be so small that estimating all of them accurately might be a hopeless task. We also move to a simulation framework that is more representative of the real Internet. In Section 6.3.1, we compare CDF estimates (for the simple simulations used in the previous section) obtained using the 3 estimators of $C(l)$. We find that the estimator based on the weak assumption is undesirable because it does not estimate a normalized CDF. This further justifies our shift to CDF estimation because it incorporates natural constraints

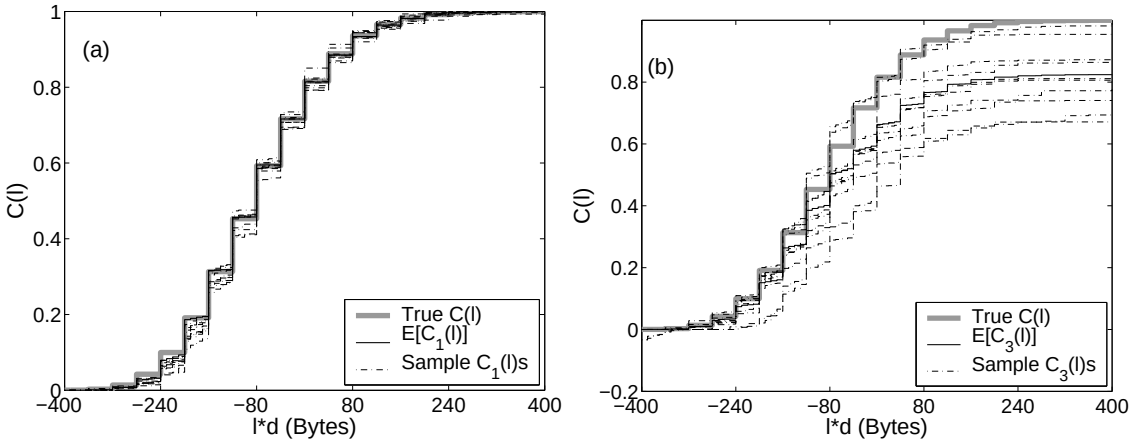

Figure 6.7. Comparison of CDF estimates (solid dark line) and true CDF $C(l) = P(C \leq l)$ (thick grey line). (a) $\hat{C}_1$ with $\mathbf{r}(l) = 4p/d$, expectation and 8 samples, bias and variance are small with $\rho = 0.8$ but depend on l; (b) $\hat{C}_3$, with $\mathbf{r}(l) = 4p/d$, estimates are individually worse and not naturally normalized.

such as normalization. We also follow up on our discussion on the bias-variance trade-off by illustrating the advantages of choosing an adaptive $\mathbf{r}$ for CDF estimation. To investigate adaptive algorithms, we need to use more realistic simulation settings. We describe them in Section 6.3.2. Our use of CDFs requires us to redefine the strong assumption for CDFs (as opposed to density). We do this in Section 6.3.3.

6.3.1 From Density to Distribution

To consider estimates of the CDF $C(l)$, we define CDF estimators, based on the previously-defined $c(l)$ estimators, as follows

$$\hat{C}_j(l) = \sum_{i=-t}^{l} \hat{c}_j(i), \tag{6.6}$$

for each of $j = 1, 2, 3$, rather than examining the density estimates $\hat{c}_j(l)$ directly. To complete the above definition, we must also specify the parameter values $\mathbf{r}(l)$ used for each l. Note, by the definition above and that of $\hat{c}_1$ (Equation 6.1), we have $\hat{C}_1(l) = \hat{G}_{\mathbf{r}}(l + x)$.

Figure 6.7(a) compares $\hat{C}_1(l)$ with $\mathbf{r}(l) = 4p/d$, again based on $n = 1000$ probes, against $C(l)$, using the same cross-traffic as in Section 6.2.3. The expected CDF (calculated as the average of $N = 1000$ experiments) is very close to the true CDF, and the variance, illustrated informally by the plotting of 10 individual samples, is likewise small. Both however are functions of l: the bias is

greatest at small l, whilst the variance is larger at intermediate values. The lower bias at larger l is easy to understand from Figure 6.2 given how the diagonal $\mathbf{r}(l) = 4p/d$ moves to the right of where $h(k,l)$ is significant, allowing the strong assumption to hold. Note that, since $\hat{c}_1(l) = \hat{g}_{\mathbf{r}}(l+x)$, and $\mathbf{r}(l)$ is constant, the natural normalization of $\hat{G}_{\mathbf{r}}(u)$ is naturally transferred to $\hat{C}_1(l)$. Thus, even though estimates of $\hat{c}_1(l)$ must eventually be poor when l is very large, this does not prevent good behavior of the CDF. This natural normalization is an extremely desirable property.

Figure 6.7(b) offers exactly the same comparison as in plot (a), only for $\hat{C}_3(l)$. Because of the correction terms in Equation (6.3), $\hat{c}_3(l)$ does not possess the natural normalization enjoyed by both $\hat{c}_1$ and $\hat{c}_2$. Consequently, the errors in the density (apart from being individually considerably worse as discussed above) are not constrained to cancel at large l in the same way, leading to CDF estimates with fundamentally flawed properties. As a result, $\hat{c}_3(l)$ will not be considered further.

From this point on, we omit results for $\hat{C}_2$ too, concentrating solely on $\hat{C}_1$. We do this mainly because the results for the two are very similar once we go to systems with more realistic packet sizes and smaller levels of discretization. This is because, under realistic conditions with small discretization, either 0 or 1 samples belong to any given angle set. Consequently, the weights appearing in the definition of $\hat{c}_1$ become, in a sample path sense, uniform, just like those of $\hat{c}_2$.

Since $\rho = 0.8$ in Figure 6.7(a), there is enough data to provide good estimates at most l values of significance. Figure 6.8 shows how the performance of $\hat{C}_1(l)$ drops significantly when $\rho = 0.4$ (expectation estimates only are shown, again based on $N = 1000$). The l dependence of the bias is now strong and clearly visible, as is the dependence on the choice of $\mathbf{r}$. When moving from $\mathbf{r} = 3p/d$ to the lower diagonal of $4p/d$, the bias becomes even worse at small l but improves markedly at large l. This suggests that an adaptive strategy, where $\mathbf{r}(l)$ truly depends on l, could be used improve estimation. An example of this is given, where a transition from $\hat{C}_1(l)$ with $\mathbf{r} = 3p/d$ to $\hat{C}_1(l)$ to $\mathbf{r} = 4p/d$ occurs at $l = -4p/d$ (i.e., 6 packets arriving between probes).

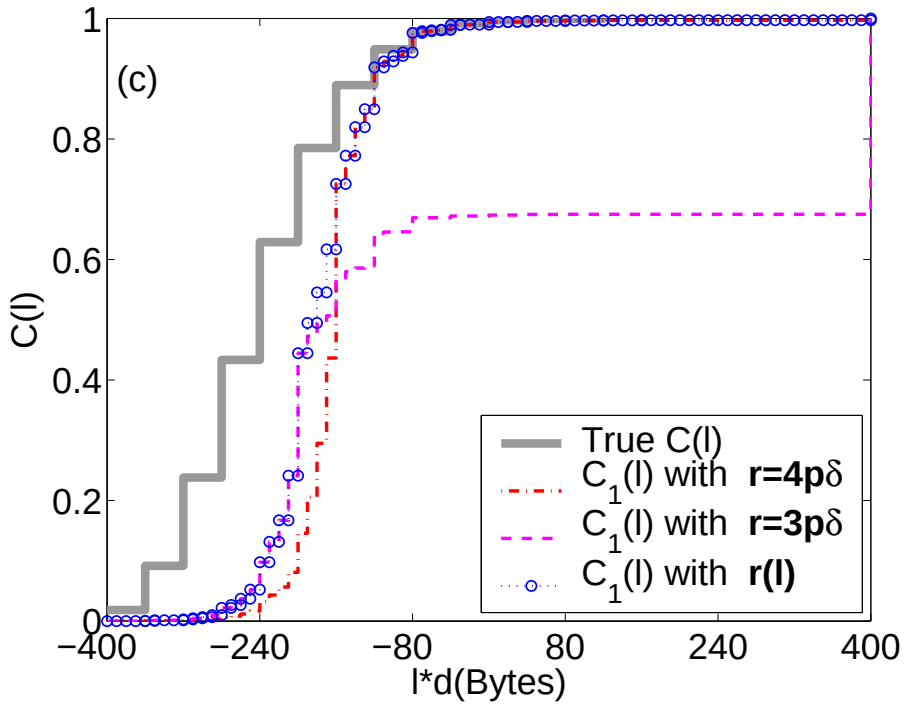

Figure 6.8. Comparison of CDF estimates $\hat{C}_1$ with three different choices of $\mathbf{r}(l)$. The true CDF is the thick grey line. Varying $\mathbf{r}$ with l performs better than either of the best static choices $\mathbf{r}(l) = 3p/d$ and $4p/d$.

6.3.2 Experimental Setup

We abandon constant packet sizes entirely from now on, and move to the following trimodal packet size distribution which is similar to that found in the Internet[Spr]:

$$p(i) = \begin{cases} 0.5, & i = 40; \\ 0.1, & i = 580; \\ 0.4, & i = 1500; \\ 0, & \text{otherwise}, \end{cases} \tag{6.7}$$

The cross-traffic arrival process remains Poisson, i.e., an interval t consists of a Poisson distributed number of packets. This is a *compound Poisson* distribution, which is much more bursty when $p(i)$ is not concentrated on a single value (the constant packet size case). We calculate the CDF of this distribution by using a numerical approximation of an exact formula for $c(l)$, being simply the Poisson-weighted (with parameter λt) sum of terms, where the j^{th} term is the j-fold convolution of the density $p(\cdot)$ above. The point-wise error can be controlled and was chosen here to be 10^{-5},

negligible compared to other factors. We use $n = 1000$ probes of 40 bytes (conveniently, all packet sizes integral multiples of the discretization unit $d = 10$). We denote p_{max} to be the maximum size of a packet, 1500 bytes and x_{max} to be its transmission time on the link. Also, we change the link capacity from 320Kbps to a realistic 10Mbps.

6.3.3 Redefining the strong assumption

Since we moved to estimating CDFs, we need to redefine the strong assumption in terms of the CDF of $c(l)$, $C(l)$. We do this below. We know that $C(l) = H(k, l)$ for $k \geq l + t$. We approximate this by $H(\mathbf{k}, l)$ for $\mathbf{k} \leq l + t$, thereby potentially ignoring the part of the density nearest the right hand edge of the strip. Consider the lower angle in Figure 5.2. If we approximate $C(l_2)$ by $F_{r_2}(s_2)$, we are ignoring the density in a triangular region consisting of two squares at height l_2 and one square at height $l_2 - 1$. Formally, we create an error

$$e_s(\mathbf{k}; l) = H(l + t, l) - H(\mathbf{k}, l) \geq 0 \tag{6.8}$$

corresponding to the sum of $h(k, l')$ over a triangular region defined by $k > \mathbf{k}$, $l' \leq l$, and the lower boundary of the strip. Such a definition is different to what we would obtain if we assumed the strong definition of Equation 5.26 over the range $l' \leq l$. A minor difference is that the new definition includes the density lying strictly below the corner $(\mathbf{k}, l)$, which was previously excluded. The larger difference is that the previous definition implied that, for any l, the estimate of $C(l)$ would assume zero mass in a region adjacent to the lower edge of the strip for **all** $l' \leq l$. In contrast, the new definition neglects only the mass in a triangular region described, and does not enter $k < \mathbf{k}$. Thus, the new definition demands progressively weaker conditions as l increases, compared to the previous one.

We define the *strong assumption curve* at probability threshold θ as follows:

$$k_s(l; \theta) = \max(0, \arg\min_{k \geq 0} e_s(k; l) < \theta) \tag{6.9}$$

that is, for each l, the leftmost k value within the strip such that the error due to the strong assumption does not exceed θ. Figure 6.9 gives examples (estimates made using $N = 1000000$) of the strong assumption curve for three threshold values at high utilization.

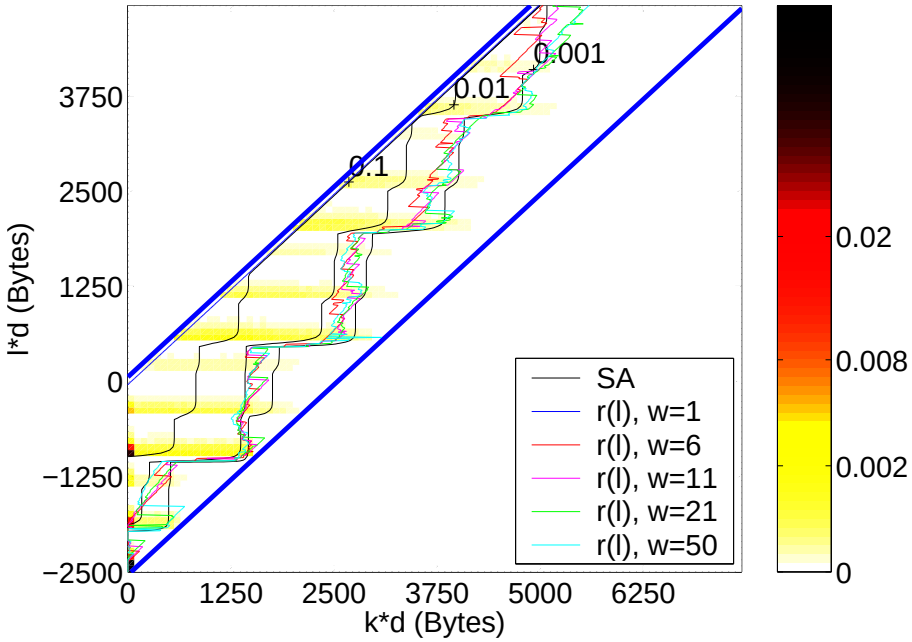

Figure 6.9. Examples of $C(l)$-based strong assumption curves for $\theta = \{0.001, 0.01, 0.1\}$ (3 grey curves) for $\rho = 0.8$. Estimates from the saturation algorithm are also shown for different window sizes w.

For some θ and l, let $\mathbf{k}$ be $k_s(l, \theta)$. Then, $H(l + t, l) - H(k, l)$ is at most θ for $k \geq \mathbf{k}$. This is because the rectangles $H(k, l)$ extend beyond $H(\mathbf{k}, l)$ and must be progressively better approximations. Hence, $C(l) - F_r(r + l + x) \leq \theta$ for all $r \geq \mathbf{r} = \mathbf{k} - l - x$. So, any weighted combination of these $F_r(r + l + x)$ (with weights summing to unity) would differ from $C(l)$ by at most θ. In particular, $C(l) - G_{\mathbf{r}}(l + x) \leq \theta$. Thus, using $\hat{G}_r(l + x)$ to estimate $C(l)$ is consistent with the new definition too.

6.4 Final Composite Estimators

In this section, we define our final estimators. First, in Section 6.4.1, we propose a window-based *saturation algorithm* to choose $\mathbf{r}$ adaptively as a function of l. We find that ensuring the monotonicity of an estimator using an adaptive $\mathbf{r}$ is not easy. We explore multiple algorithms to enforce monotonicity and build a composite estimator incorporating the saturation algorithm and these monotonicity algorithms in Section 6.4.2. In Section 6.4.3, we explore a different kind of

adaptive **r** estimator that automatically enforces monotonicity. We evaluate these various "final" estimators in later sections.

6.4.1 Estimating the strong assumption curve

Our guiding principle to choose $\mathbf{r}(l)$ follows from the observations of the previous section: if $\mathbf{r}(l)$ can be chosen to match where the strong assumption begins to fail, essentially tracking the boundary where the density in Figure 6.2 drops off, then the bias-variance trade-off observed in Figure 6.6 could be well managed for each l. However, this approach will fail when there is no data to measure the position of this boundary, as for example in the left plot of Figure 6.4. In this low ρ plot, the density $h(k,l)$ of (B,C), roughly speaking centered about $(k,l) = (0,-t)$ (note the tiny black region), is well separated from the density of available data, centered about $(k,l) = (0,-x)$, corresponding to $(r,s) = (0,0)$, i.e., with high probability the probe delays are close to the minimum.

Figures 6.10 and 6.11 plot $\hat{G}_r(l+x)$ against r (rd [bytes]) for four l values, for low and high utilization respectively. The noisy curves are several independent estimated functions each based on a single realization of $n = 500$ probes, whereas for comparison, the thick grey curve derives from a realization employing 1 million probes. As r increases, $G_r(l+x)$ increases monotonically to attain $C(l)$ (shown as the horizontal line) at $r = t - x = 246$ (the full height vertical line). The estimates roughly follow this pattern. However, since the available data monotonically decreases with r, the crucial limiting behavior becomes obscured by noise which can take extreme values. Moreover, the curves show non-ergodic features in that they oscillate about a limiting level which is not necessarily $C(l)$ but some random offset from it. As a result, it is not feasible to target the point lying on a strong assumption curve given by a small threshold θ.

Therefore, we adopt a less ambitious approach which aims to find the point at which the steadily increasing phase of the estimate curve saturates. We first smooth the curve to reduce the sample variability, so that the systematic increase at small r can be seen more clearly. The intuition is that when the underlying "expected" curve has saturated, then the variability will cause the curve to cease to become monotonic despite the smoothing.

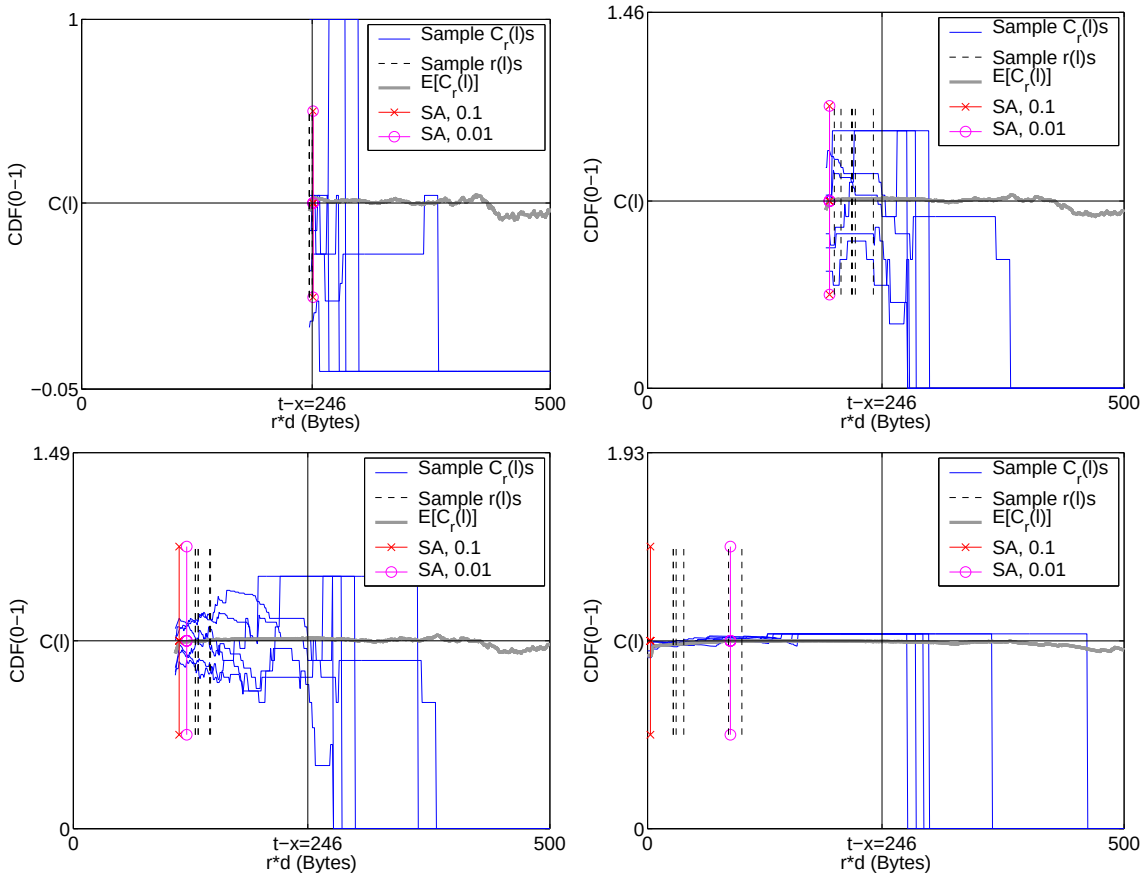

Figure 6.10. Strong curve estimation with $\rho = 0.2$ for (left to right, top to bottom) $l = \{4, 60, 140, 300\}$. The 6 thin lines (resp. single thick grey line) are estimates of $H(\mathbf{k}, l)$ using 500 (resp. 1 million) probes. The $\mathbf{r}$ values selected by the saturation algorithm are shown as vertical dashed lines, can be compared to the strong assumption values $k_s(l;\theta)$ for $\theta = \{0.001, 0.01, 0.1\}$.

Saturation Algorithm:

1. Select a window size w (performance insensitive to value).
2. Smooth the $\hat{G}_r(l+x)$ estimates using a moving average window filter of width w (the filter is causal, thus there is an edge effect over the first $w-1$ values).
3. If $r = t - x$, set $\mathbf{r} = t - x$ and exit, else set $\mathbf{r}$ to the first r for which the smoothed curve ceases to be non-decreasing [2].

The algorithm is guaranteed to terminate with a value $0 \leq \mathbf{r} \leq t - x$. Note that for $l < 0$ the

[2] In the implementation, it was not necessary to actually smooth, but only to see if the new window element entering on the right is smaller than the one departing.

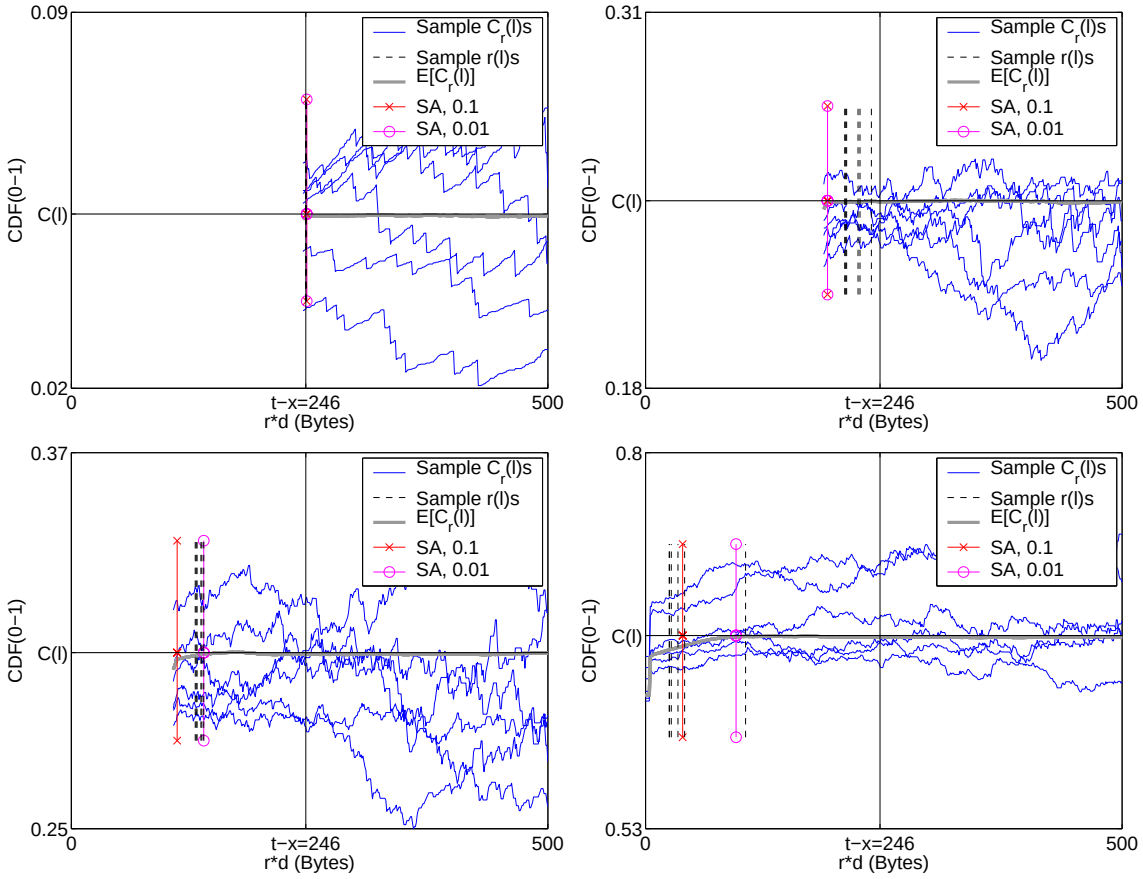

Figure 6.11. Strong curve estimation with $\rho = 0.8$ for (left to right, top to bottom) $l = \{4, 60, 140, 300\}$. The 6 thin lines (resp. single thick grey line) are estimates of $H(\mathbf{k}, l)$ using 500 (resp. 1 million) probes. The $\mathbf{r}$ values selected by the saturation algorithm are shown as vertical dashed lines, can be compared to the strong assumption values $k_s(l;\theta)$ for $\theta = \{0.001, 0.01, 0.1\}$.

minimum $\mathbf{r}$ value is constrained by the shape of the strip. In Figure 6.10 and 6.11 values are only plotted from the first entry into the strip onwards. The full height vertical line marks $r = t - x$.

As noted above, it is not feasible to locate the point $k_x(l;\theta)$ on a strong assumption curve for a given θ. By comparing the algorithm outputs in Figures 6.10 and 6.11, given by the thin half height vertical lines, against the thicker vertical lines corresponding to $k_s(l,\theta)$ for two values of θ, we see that the saturation algorithm outputs indeed do not track any particular θ value. Indeed, the algorithm is influenced not only by the distance to the saturation level $C(l)$, but also by the variability of the curves, which trigger the algorithm to exit once they become too severe in the downward direction. Thus, in some informal sense, the algorithm is performing a trade-off between bias and variance rather than being concerned solely with the strong assumption curve (which would

correspond to emphasizing the bias only). Although this means that, unfortunately, the algorithm performance cannot be tested in isolation by comparing against a target θ, it is in other respects entirely appropriate, since as already noted, $k_s(l;\theta)$ may be inherently impossible to measure without bias if the coverage of h is poor. On the other hand, the expected $\mathbf{r}(l)$ curves (projected into the (k,l) plane) of Figure 6.9, estimated by using a million sample simulation, shows that the algorithm does on average output a function which roughly correspond to a strong assumption curve (in this case with $\theta \approx 0.001$) as originally intended. This graphs also illustrates the fact that $\mathbf{r}(l)$ generally decreases with l, following the strong assumption curve.

Figure 6.9 also shows the important property of insensitivity of the algorithm with respect to the window size parameter larger than 1. Here, d is 10 bytes and t is 250 slots. Hence, $w = 25$ (slots) corresponds to $t/w = 10$. Since the algorithm performance was good as long as w was neither too close to 0 or t, we use a default value of $t/w = 10$. Later, we also justify this choice by comparing performance of CDF estimation with various window sizes.

6.4.2 Defining the Composite Estimator

They are many possible ways in which one could make use of $\mathbf{r}(l)$ to design new estimators. The example in Figure 6.7(c) at the end of Section 6.3.1 simply moved from one constant-**r** CDF estimator to another at a particular value of l. This simple approach in fact enjoys an important property. This is the fact that, since constant-**r** estimators are naturally normalized (i.e., they tend to 1 at large l), so is the new adaptive estimator, and this extends immediately to arbitrary $\mathbf{r}(l)$.

One can therefore define a $\mathbf{r}(l)$ based estimator as follows. First, calculate $\mathbf{r}(l)$ as in the previous section. For each of the different r values appearing in the function, calculate the corresponding estimator $\hat{C}_1(l) = G_{\mathbf{r}(l)}(l+x)$. The idea is that by moving between members of this 'bank' of constant-**r** estimators, we can obey $\mathbf{r}(l)$ whilst simultaneously preserving natural normalization. This property would **not** have been achieved if instead, for example, we had tried to adapt the density estimate instead of the CDF, say by setting $\hat{c}(l) = \hat{g}_{\mathbf{r}(l)}(l+x)$.

The disadvantage of the above naive or *raw composite* method is that there is no guarantee that the resulting CDF is monotonic. The CDF could move downward when we switch to a new member

of the bank. Thus, although it could be used to estimate a given fixed quantile, it is not useful for measuring the probability of smaller sets, as it may assign negative probability to them. We investigate three methods which modify the raw composite estimator to form a *monotonic composite* estimator whose sample functions are both normalized and monotonic.

Monotonicity Algorithms applied to raw composite CDF $C(l)$

- **Left-to-Right (L2R)**: Move left to right, forming $C'(l) = \max(C(l), C'(l-1))$.
- **Right-to-Left (R2L)**: Move right to left, forming $C'(l) = \min(C(l), C'(l+1))$.
- **Data Pinning**: Obtain the number n_l of probes with $R \geq \mathbf{r}(l)$, initialize a set Q of processed l values to null. In order of decreasing size of n_l, recursively assign $C'(l) = C(l)$, then ensure its consistency (enforce monotonicity) at all l values already in Q, then add l to Q. More precisely: $\forall l' \in Q, l' < l$, set $C'(l) = \max(C'(l), C'(l'))$ and $\forall l' \in Q, l' > l$, $C'(l) = \min(C'(l), C'(l'))$.

The data pinning algorithm "pins" the estimate at the l values which have the most available data whilst enforcing monotonicity, proceeding recursively between the pinned values until the entire function is determined. This corresponds to a kind of constrained intrapolation, weighted by available data.

Figure 6.12 gives an example of a raw composite estimate based on $\mathbf{r}(l)$, together with the three monotonicity enforcing algorithms just described. The upper curves are the expected CDF functions, obtained by averaging over $n = 1000$ independent experiments. The lower curves show the standard deviation of the same estimates as a function of l (using the same vertical scale). The raw curve shows a reasonably small bias at all l, and a clear lack of monotonicity. Although this may be due to estimation error of the expected curve, individual sample functions are not monotonic. As expected, L2R makes the estimate move up, and R2L makes it move down. Perhaps unexpectedly, Data Pinning shows results which are almost indistinguishable from R2L. Since r(l) mostly decreases with increasing l, R2L and data pinning algorithms perform close to each other. This is also true for the standard deviation of the two, which is better than that of raw and L2R. In all cases however, the monotonicity algorithms create significant differences in bias, for small

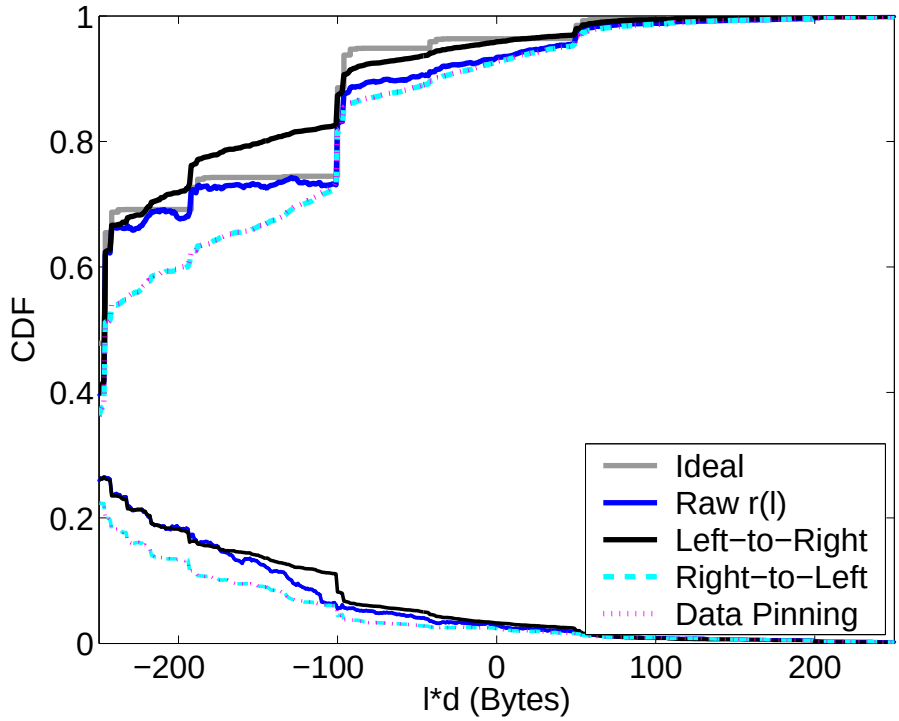

Figure 6.12. Statistics of composite estimators $\hat{C}_1'(l)$ based on $\mathbf{r}(l)$ with $t/w = 10$ ($t = 250\delta$ and $\rho = 0.2$). The raw and three monotonic composite estimators are shown. Monotonicity causes a large bias (compare upper expected curves with the true CDF in grey) which is not compensated by a corresponding decrease in standard deviation (lower curves).

changes in standard deviation. Bias is the main problem introduced by the need for an algorithm for monotonicity.

Three sample paths of each estimator, in a data poor case with $\rho = 0.1$, are shown in Figure 6.13. Because their behavior is similar, we show Data Pinning but not R2L to reduce clutter. The sample paths of the raw composite estimator are extreme. Typically they rise to 1 at small l where there is no data and hence where bias is extreme, before improving at intermediate l. At large l data is again scarce but the natural normalization property limits the absolute value of bias. The L2R estimator performs very poorly as it locks in the terrible performance at small l due to which (better) estimates at larger l cannot decrease. Data Pinning (and R2L, not shown) perform much better, but we see that their variance is considerable. It is important to note that here we are zooming in performance under very difficult conditions where there is "almost no data" for the estimator to work with. Under richer data scenarios, all these variants perform quite well.

From these results we learn that there is limited benefit from attempting to "smooth" $\mathbf{r}(l)$, as a

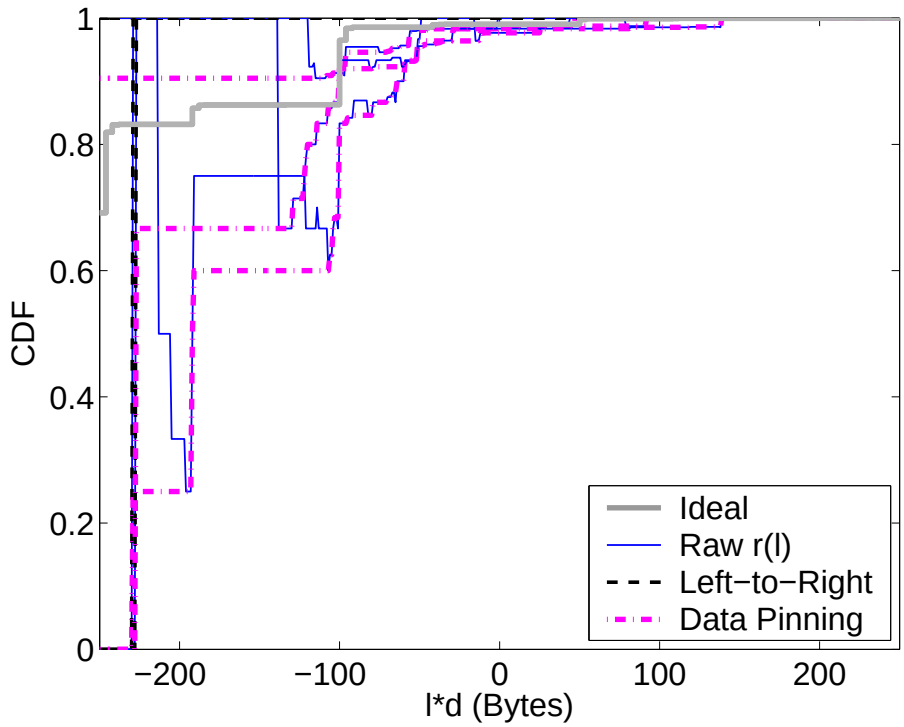

Figure 6.13. Sample Plots of Raw and Monotonized Estimates for $\rho = 0.1$ and $t = 1.67x_{max}/d$. We do not show the R2L estimates to reduce clutter. This plot illustrates how the bias and variance of per-l estimates are dependent. This dependence causes across-l metrics (sup-norm and L1-norm) to have different trends than the per-l metrics.

way of reducing the number of values that $r(l)$ takes and therefore the ill-effects of the monotonicity algorithm. Even if $\mathbf{r}(l)$ were taken to be piecewise constant with only two values, in data poor cases we may still be moving between sample CDFs which are very crude, resulting in large errors over large ranges of l values. Indeed, in scenarios where angles are so scarce that there are only j values of l where they can be found, the corresponding sample CDF will contain only $j - 1$ jumps, a very crude approximation of the true $C(l)$ which has many more. This is typical of problems found in empirical estimation of discrete (or continuous) densities from limited data.

6.4.3 An Adaptive Constant r Estimator

The monotonicity issues do not arise with the "underlying" estimators that have a constant **r**. They are attractive due to their simplicity and deserve to be explored. But, to use such a constant **r** estimator, one must select the value of the parameter **r**. In some cases one may already have a good

idea of the important parameters controlling h, particularly ρ and t, and thereby have reasonable values of the marginal of R, from which appropriate values of $\mathbf{r}$ could be tabulated. For example, some quantile r_q of R may be chosen as $\mathbf{r}$. In practice, this may not be the case. Hence, we use an adaptive estimator which selects $\mathbf{r}$ according to the following principle: to locate the edge of available data. That is, it aims to find an $\mathbf{r}$ small enough so that some data will lie below it in the strip (as it is essential that the estimators will have some data to work with), but not to go much smaller (higher in the strip) than that, in order to avoid bias.

The above principle implies that the appropriate way to measure "data available" is absolute rather than relative. Accordingly we choose $\mathbf{r} = \min\{t - x, r_*\}$, where $r_* = \hat{F}_r^{-1}((n-m)/n)$ is the r corresponding to having at least m observations. When data is plentiful $\mathbf{r}$ will default to $\mathbf{r} = t - x$, as r values to the right of the strip are always bias free. If data is scarce, this estimator will choose, for all l, the $\mathbf{r}$ that corresponds to at least m observations. Note that $\mathbf{r}$ depends on the data used to estimate $C(l)$ itself. Hence, this estimator can no longer be regarded as a constant $\mathbf{r}$ one. Instead, it is a more sophisticated adaptive one with a constant $\mathbf{r}$ value at all l. There is still a need to automatically select m. We examine performance using a range of different m values in Section 6.5.

6.5 Estimator Performance

In this section, we examine the performance of variants of $\hat{C}_1$, as defined in the previous section, as a function of cross-traffic, and of t. In Section 6.5.1, we define metrics based on Mean Square Error (MSE) that are suitable for measuring the performance of CDF estimators. In Section 6.5.2, we compare the basic estimators (without the monotonicity enforcing algorithms). We find that our adaptive algorithm to choose $\mathbf{r}$ performs better than all other estimators. In Section 6.5.3, we compare the various monotonicity-enforcing algorithms. We find that performance differences are seen only when data is neither too scarce or plentiful.

For this section, we continue to use consecutive probes of a periodic stream for quicker simulations since the joint ergodicity conditions are satisfied. The estimator variants and parameter values are:

- Constant **r**:
  - $\mathbf{r} = 0$, the naive packet pair heuristic
  - $\mathbf{r} = t - x$, pure Class 1 (low bias, high variance)
  - $\mathbf{r}$ = quantile corresponding to $m = 8$ on average, i.e., $q = 1 - m/n = .008$
- Adaptive constant **r**:
  - $m = 2$ use very little data (low bias, high variance)
  - $m = 8$ compare adaptive to constant above
  - $m = 50$ use more data (higher bias, lower variance)
- Composite estimator using $\mathbf{r}(l)$:
  - Data Pinning: the main candidate ($t/w = 10$)
  - raw: best case for composite method ($t/w = 10$)
  - R2L and L2R: for robustness comparisons

6.5.1 Metrics

We begin with Figure 6.14, where a similar representation to Figure 6.12 is given. We see that the naive $\mathbf{r} = 0$ estimator has bias so high that its variance function is small, indicating that the great majority of estimates share the same poor behavior. Using $\mathbf{r} = t - x$ in this case produces very low bias but high variance, as expected. By entering into the strip and using $\mathbf{r} = 219$ (the quantile corresponding to $m = 8$ on average), we add bias at small l, but gain reduced variance over all l as a result. The adaptive version of this estimator, using $m = 8$ in a per-estimate sense, improves the bias performance with no variance penalty. Finally, the raw composite estimator shows low bias for most l values, and lower variance at most l values, indicating that it is worth pursuing estimators of this type.

Performance results of the type shown in Figure 6.14 are too detailed to allow coverage of the parameter space governing cross-traffic characteristics. To assess estimator performance in a

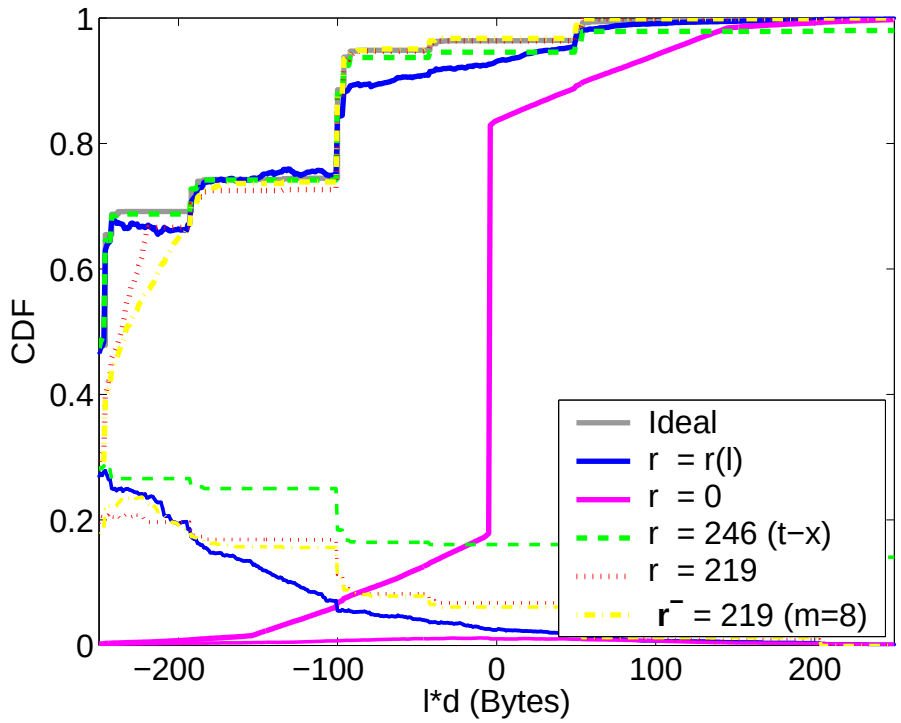

Figure 6.14. Examples of expectation and standard deviation of estimate functions for $\rho = 0.2$. Here $\mathbf{r}(l)$ is raw monotonic.

way which combines bias and variance, and examines an entire CDF, we define the following two measures, each of which returns a single number to evaluate a given sample function.

- Sup: $\mathcal{E} = \sup_l |\hat{C}(l) - C(l)|$
- L1: $\mathcal{E} = \frac{1}{l_q+t+1} \sum_{l=-t}^{l_q} |\hat{C}(l) - C(l)|$,

where l_q the qth quantile of $C(l)$.

The first of these measures the worst departure from the true CDF over all l, whereas the second gives a measure of the average departure. We cannot let $l_q = \infty$, as this would be identically zero for any two distributions, no matter how different, due to the domination of the tail where $C(l) \approx 1$ out to infinity. Instead, we assess the degree of difference only over the main body of the distribution by using $q = 0.95$.

For each measure the random variable $\mathcal{E}$ takes values in $[0, 1]$. Our performance metrics are the Mean Square Error (MSE) , defined as

$$MSE = \mathbb{E}[\mathcal{E}]^2 + \mathrm{Var}[\mathcal{E}], \tag{6.10}$$

of the corresponding measures. We use the square root of MSE, the Root Mean Square Error (RMSE). These also take values in $[0, 1]$. The expectation and standard deviation of $\mathcal{E}$ are estimated, in the usual way, using $N = 1000$ independent experiments, and are of interest in their own right, as components of the MSE.

6.5.2 Basic Estimators

Results are given in Figure 6.15 for the (root) MSE using the Sup and L1 measure. The three plots sample the (ρ, t) parameter space which controls h and therefore C. Figure 6.15(Top) shows performance as a function of ρ for a fixed t. We see that the naive estimator $\mathbf{r} = 0$, which blindly applies the packet pair heuristic, performs very poorly, whereas the Class 1 estimator with $\mathbf{r} = t - x$ performs almost as well as the sophisticated variants once ρ exceeds 0.2. This indicates that for these (ρ, t) combinations there is sufficient data, and the methods are effectively defaulting to using $\mathbf{r} = t - x$. We see a steady improvement as ρ increases, since increasing data leads to lower variance and hence MSE. The same general results hold for the Sup and L1 norms. Of course, the root MSE for the L1-norm is smaller than the Sup-norm root MSE.

Figure 6.15(Middle) shows the effect of increasing t by a factor of 4. The effective loss of available data sees $\mathbf{r} = t - x$ performing poorly now until ρ is 0.6, since there is little mass to the right of the strip, except at very high utilization. At $\rho = 0.1$, data is so scarce that all methods have errors which are equal because they are the worst possible, namely equal to 1 for some l. Significant improvement is achieved by using the tuned Class 2 estimators which enter into the strip, not not too far, provided that ρ is high enough. The adaptive estimators all perform well. However, the need to choose m wisely is apparent: as ρ increases, larger m performs better, although at still larger ρ, they all default to $t - x$ as so perform identically. The adaptive and constant variants of $m = 8$ perform similarly, although the adaptive one is consistently slightly better. Finally, the raw composite estimator shows uniformly good results, demonstrating a satisfying adaptivity to the amount of data available.

Figure 6.15(Bottom) shows the effect of increasing t at fixed $\rho = 0.6$ (the first and fourth t values correspond to those of the Top and Middle plots respectively). Not surprisingly, all methods

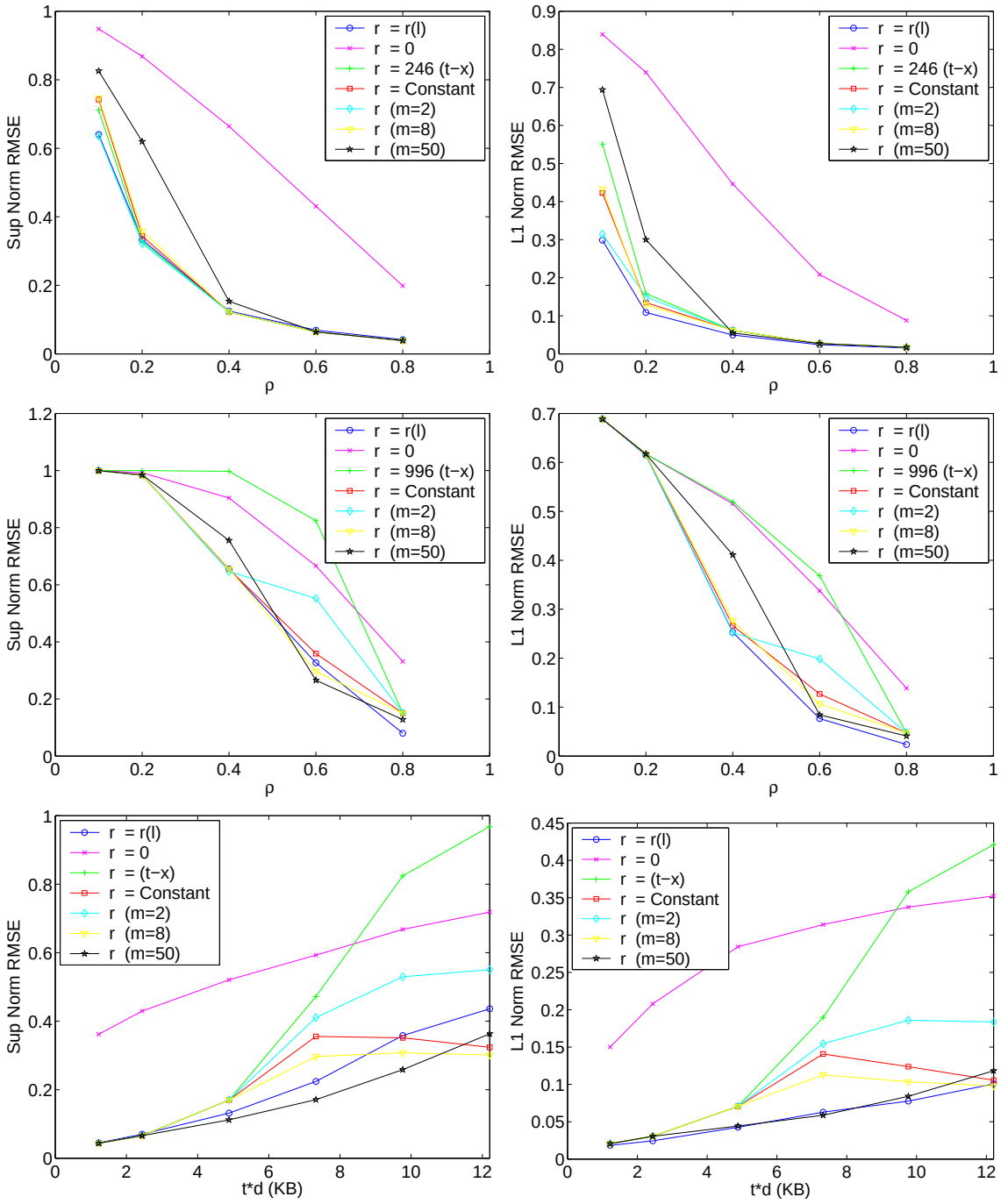

Figure 6.15. Sup-norm and L1-norm performance using trimodal packet sizes. (Top) 7 estimators as a function of ρ, $t = 1.67x_{max}/d$; (Middle) The same estimators with $t = 6.67x_{max}/d$; (Bottom) Dependence on t, with $\rho = 0.6$. Here $r(l)$ denotes the *raw* composite estimator.

perform worse at greater t, as there is effectively less data available. There are a couple of exceptions at the largest two t values. We believe that these are due to the definition of the measures which does not scale appropriately as $C(l)$ evolves with t. In particular, there is a need to exclude the tail not

only at large l but also at small l. As the left tails grows in length as t (recall that $\mathbb{E}[C]$ is proportional to t), this will reduce the measure at large enough t. There may also be some contribution due to variability in our estimation of these performance curves, which use only $n = 1000$ probes. At larger t, the cross-traffic is effectively less bursty as far as observations by probes are concerned. Just as in the case of low utilization, reduced burstiness robs the probes of the variation in delays they require to obtain data from the required region. In particular it is more difficult to find mass to the right of the strip. Although errors build quite rapidly for larger t, good estimator design has the potential to slow this growth substantially, whereas the extremes $t = 0$ and $t = t - x$ perform very badly in general. Again we find that the raw composite estimator successfully adapts to the changing traffic conditions, whereas for the adaptive estimator m must be chosen appropriately.

Overall, from Figure 6.15, we find that, for utilization larger than 0.5, the intra-pair gap can be about ten times the transmission time of the largest packet without the estimation error being more than 0.2.

6.5.3 Enforcing Monotonicity

As the raw composite estimator shows considerable promise, we now subject it and its monotonized variants to a more detailed performance study. Figure 6.16 explore the performance over (ρ, t) space in a similar way as before. In each plot, the three different monotonicity algorithms, and the raw composite, are compared, for each of three different window sizes. To avoid clutter, the results for different window sizes have been displaced horizontally using duplicate ρ axes. It is important to emphasize that these results pertain to the (root) MSE of a CDF. Thus, the l value at which the Sup is found will vary from sample to sample. In contrast, Figure 6.12 showed displayed expected results for each l fixed.

The results of Figure 6.16 show a remarkable lack of variation across both the methods and the window sizes. Part of this is understandable. For each window size, each monotonicity algorithm uses the same underlying raw $r(l)$, and so shares the same environment in terms of data availability. For very large t however we do see that windows sizes that are too large perform poorly, which can be understood by noting that when data poor, selecting smaller $\mathbf{r}$ is necessary to capture the few

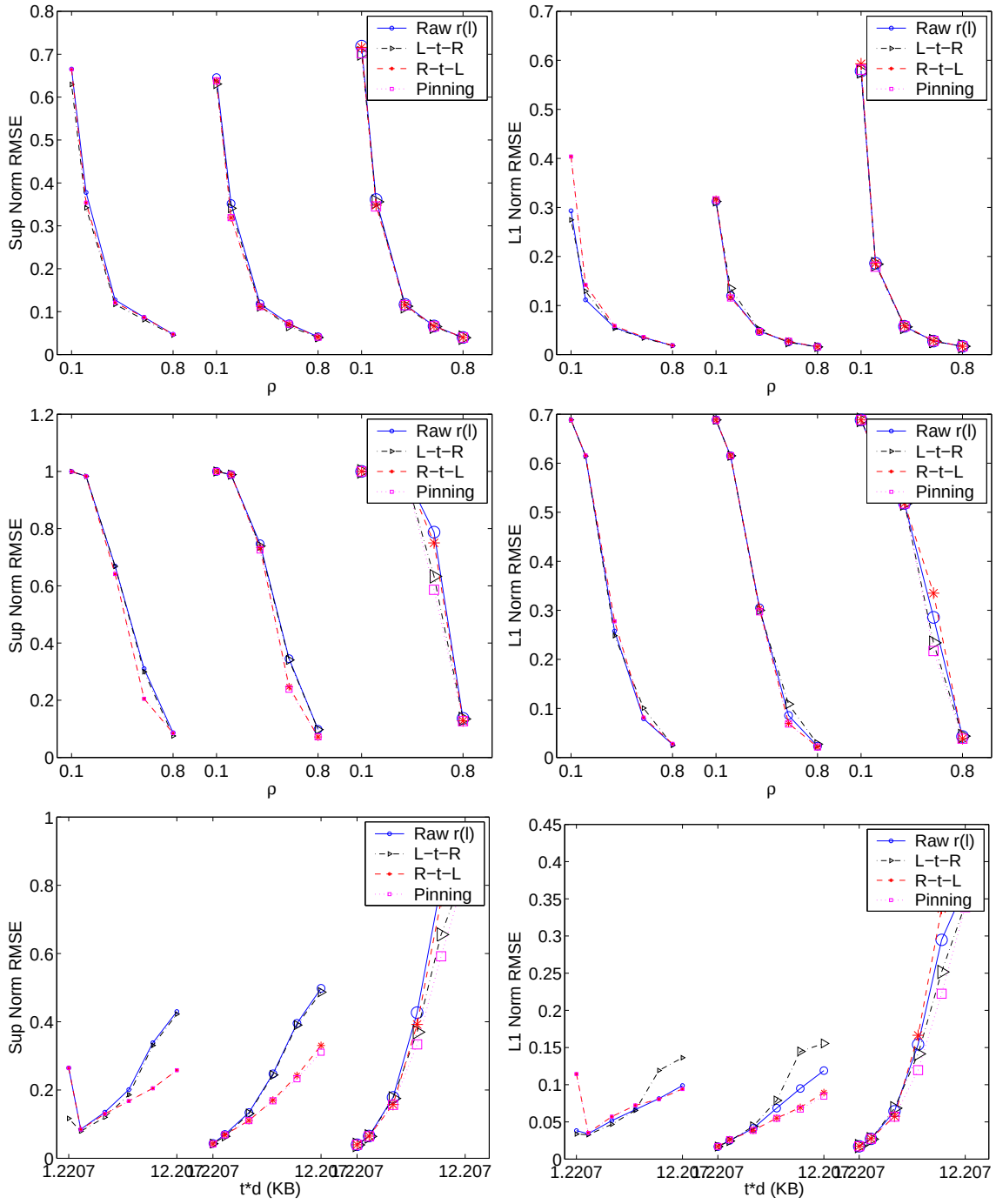

Figure 6.16. Sup-norm and L1-norm performance of various composite estimators. The ρ access is repeated for three window sizes: $(t/50, t/10, t/2)$. (a) As a function of ρ, $t = 1.67x_{max}/d$; (b) As a function of ρ, $t = 6.67x_{max}/d$; (c) Dependence on t, with $\rho = 0.6$.

angles that are available, whereas larger w will favor the algorithm triggering at larger $\mathbf{r}$. We also find that when there is a difference, Data Pinning performs best among the monotonic estimators, as we might expect from the insights of Figure 6.12. In fact, it performs even better than the

raw composite, although the differences are typically small. This apparent contradiction can be explained by noticing that looking at ensemble performance with l fixed, or not, corresponds in fact to two very different metrics which are not obliged to correspond. We hypothesize that, although at a fixed l raw is clearly superior (recall Figure 6.12), sample functions are so variable that good behavior at some l is systematically compensated by worse behavior elsewhere, resulting in a final performance which is less dependent on the details of the monotonicity algorithm than one might have supposed. This is good news in the sense that the low bias of the raw composite estimator can effectively be achieved in a monotonized version, and in particular, the results of Figure 6.15 for the raw composite, which correspond to the central plots in Figure 6.16, still hold for the monotonic variants, especially Data Pinning. Finally, we note that L2R performed marginally better in some data rich cases (small t and large ρ) not shown here.

To summarize, in data poor cases, sample functions can be extremely crude and large errors are made by all methods, for example sample CDFs containing only 1 or 2 jumps, which could have a Sup error of 1. This is not apparent when collecting per-l statistics over large numbers of experiments as in Figures 6.14 and 6.12. On the other hand, if data is plentiful, then the sample functions are relatively detailed and almost monotonic from the beginning, and so the effect of the different algorithms is not very large. Thus, the differences between algorithms and methods manifest themselves in the intermediate zone where data is neither too scarce, nor plentiful.

6.6 Trace Analysis

In this section, we use real data from core Internet routers to demonstrate the performance of our estimators with real traffic. In Section 6.6.1, we use passively-collected traces of cross-traffic to compare the various estimators. In Section 6.6.2, we use a novel experimental setup involving simultaneous passive capture and active packet injection. Our results confirm that our estimators perform very well in real world conditions. For instance, with link utilization's that are as low as 0.5, the CDF estimates have an (RMSE) error of less than 0.2 with an intra-pair gap that is tenfold the time taken to transmit the largest packet (of size 1500) bytes. We also found that our estimation of the joint distribution of (B, C) was (visually) similar to the true distribution.

6.6.1 Trace-driven Simulations

In the first set of experiments, we used real cross-traffic traces from the full router experiment describe in Chapter 3 (see [HVPD04] too). This experiment recorded all packets entering every interface of a router in a large tier-1 ISP over a 13-hour time period. We modeled the output buffer of a particular OC-3 output interface that had a reasonably high utilization, and replayed the cross-traffic that passed through it. The arrival times of this cross-traffic was taken to be the (recorded) arrival times to the router. The link was simulated as a FIFO queue.

Data Overview

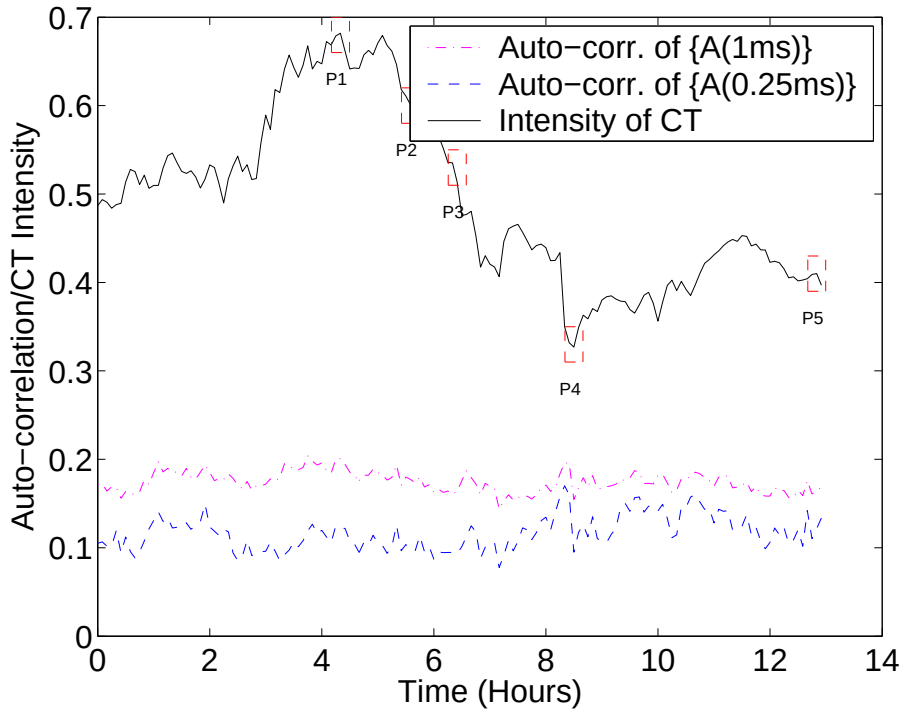

Figure 6.17. Traffic characteristics at the chosen OC-3 link. The solid black line shows the byte intensity measured over 1 second intervals. Five 5 minute intervals with a spread of ρ values from 0.33 to 0.68 were identified. The dashed and dotted curves are the auto-correlation estimates, based on looking at $A(t)$ in consecutive periods of duration $t = 1$ms and $t = 0.25$ms, calculated over 5 minute intervals over the entire duration of the trace.

By using these traces, we do more than make use of a source of realistic cross-traffic. Because of the complete monitoring, fine-grained detail of all input packets destined to the chosen output

link were available. It was therefore possible to reproduce the true (B, C) values, and compare them to those predicted by the estimator $\hat{h}(k, l)$.

As they are real traces, we did not control the ρ values in the traces a priori. To obtain a spread of values, we first observed the actual traffic intensity, averaged over 1 second intervals, of bytes to the OC-3 link. As shown in Figure 6.17, these range from $\rho = 0.3$ to 0.7. Five 5 minute portions of the trace, identified in order of decreasing ρ as $P1$, $P2$, $P3$, $P5$ (note the change in order) and $P4$, were chosen to provide a spread across this range. These had respective ρ values of 0.68, 0.60, 0.51, 0.41 and 0.33. We used the cross-traffic arrivals corresponding to these sub-traces to drive the simulations. The number of packet pairs that can be obtained with rare probing is not large enough due to the limited duration of the sub-traces. Hence, we use periodic probes to obtain packet pair observations in this section. Thus, our results may potentially include errors due to non-ergodic behavior and hence, performance with rare probing must be better than what we show. We used probes of size 40 bytes.

We made an attempt to measure the degree to which the cross-traffic obeys the independence assumptions at each of $t = 0.25$ms and $t = 1$ms. We did this by first splitting the entire 13-hour trace into five minute sub-traces. For each sub-trace, we generated a time series corresponding to the number of cross-traffic bytes that arrived in non-overlapping intervals of width t. We then estimated the 1st lag auto-correlation coefficient for this time series. The motivation for this is that our independence assumption is true if the cross-traffic measure $A(I)$ has independent increments in consecutive intervals of duration t. Moreover, independent increments also implies our assumption is valid with periodic probes too. The auto-correlation test above does not guarantee independence but provides a good sense of whether the independence assumption is justifiable. The auto-correlation results for the full router trace are shown as the bottom curves in Figure 6.17. We see that the auto-correlation is small over the trace and is reasonably *consistent* with our assumption. Not surprisingly, it holds better for the larger value of $t = 1$ms.

Figure 6.18 plots the distribution of excess packet delays, experienced by the trace traffic alone, for $P1$ to $P5$. There is a considerable spread across the different sub-traces, and therefore the delays experienced by probes in the corresponding experiments will likewise differ, resulting in different available data and thereby different estimator performance. To get a feeling for the trace

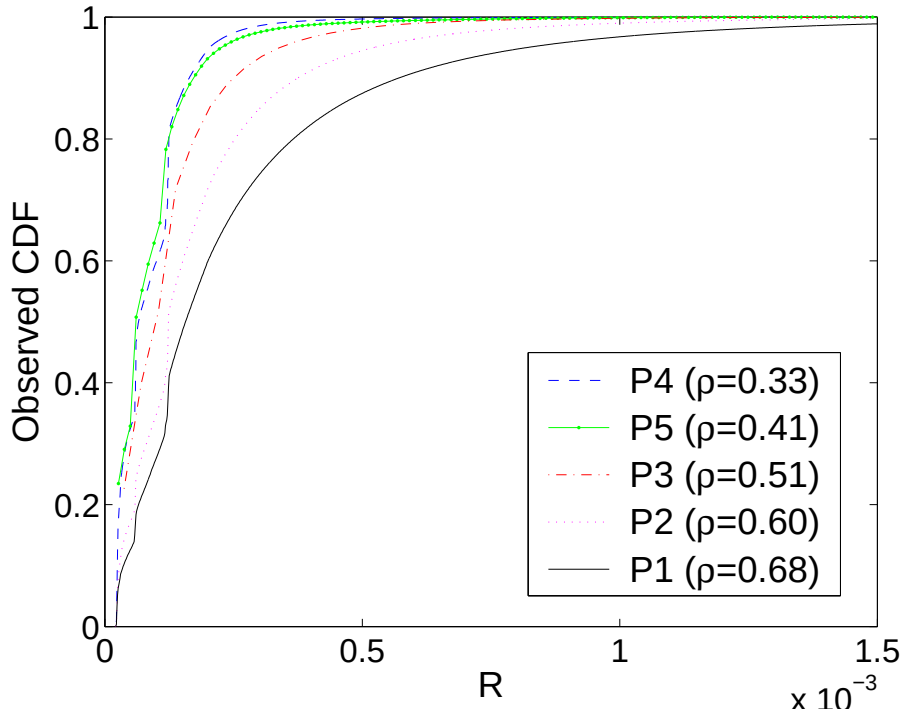

Figure 6.18. Marginals of R for the trace portions of Figure 6.17.

data, note that the quantile $r_{0.99}$ is greater than 0.25ms in all cases, and not more than 1ms except in $P1, P2$. This may be compared with x_{max}, the transmission time of a packet of the largest size ($p_{max} = 1500$ bytes), 77μs.

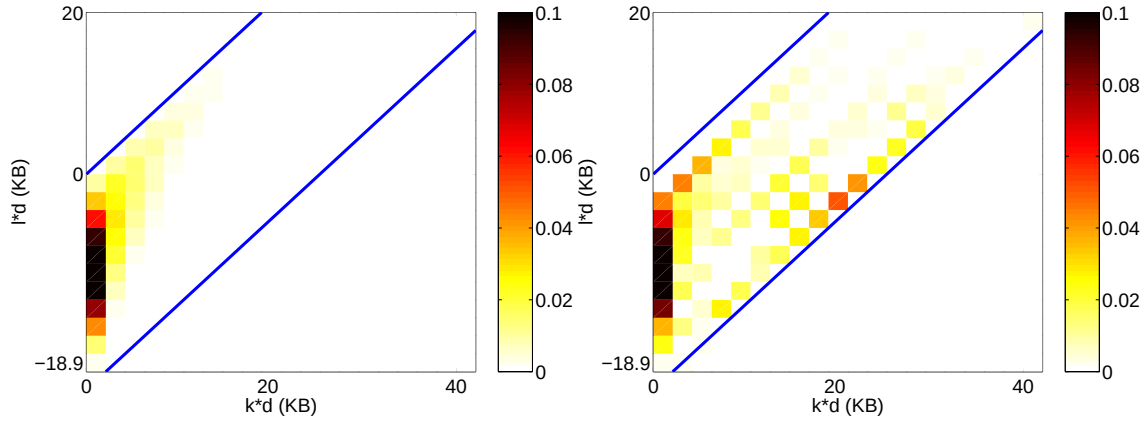

Figure 6.19. Comparison of measured and estimated $h(k, l)$ for $P1$ ($300,000$ probe packets), with $d = 10$ and and plotting resolution of $5d$ or 0.1ms for $t = 12.9x_{max}/d$ $(0.001s)$.

Figure 6.19 shows a comparison of estimated and measured $h(k, l)$ for $t = 1$ms and the sub-trace with the highest cross-traffic intensity. This corresponds to a 1000 packets per second and an

additional load of 320Kbps. Despite the fact that the estimation of a 2-dimensional distribution is inherently difficult, (although the density is plotted at a resolution above that of δ) it is clear that the essence of the cross-traffic behavior is being captured by our estimator. Densities in the middle and bottom of the strip are inaccurately estimated due to very little available data. In practice, such inaccurate estimates are easily identified since the telescopic nature of $\hat{h}(k,l)$ causes adjacent densities to be negative (we plotted them as 0)!

Performance

Examples of individual estimates are given in Figure 6.20 to show the effects of ρ and the number of probes n. As we can see, each has a dramatic impact on the ability of the estimator to recover the true CDF. The utilization has the largest impact, and can be thought of as controlling the overall bias since it affects the range of R values seen. Increasing n improves the reproduction of the CDF structure, thereby reducing error in a given sample, or alternatively provides more data which decreases variance.

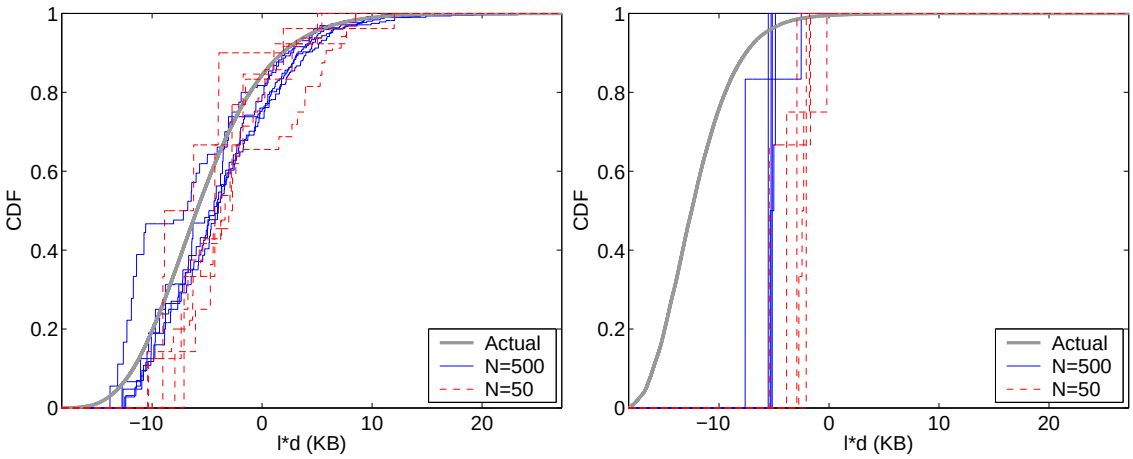

Figure 6.20. We plot sample estimates with different number of probes n for for utilization 68% and 31%, with $t = 12.9x_{max}/d\ (1ms)$.

Figure 6.21 shows the analogous results to Figure 6.15 using $n = 500$ and a discretization level of $d = 10$ bytes per slot. Here the t values are larger than those used before. The relative performance across estimators is similar to that seen earlier and the absolute performance is reasonable at the values of ρ available. The naive ($\mathbf{r} = 0$) and Class 1 ($\mathbf{r} = t - x$) estimators are both not suitable

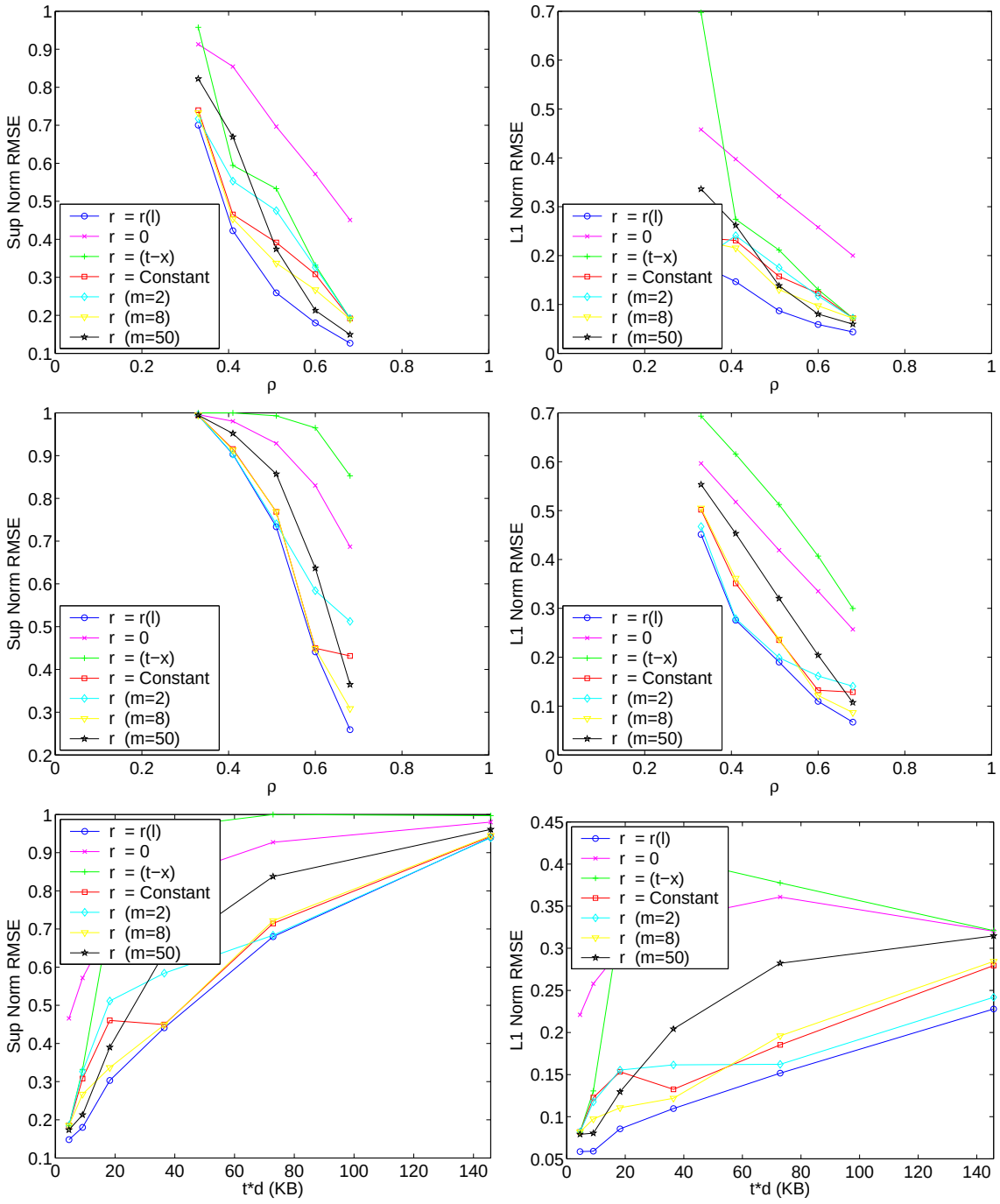

Figure 6.21. Estimator Sup-norm and L1-norm performance using router traces. We use data pinning monotonic algorithm with r(l). (Top) As a function of ρ, $t = 6.45x_{max}/d$ $(500\mu s)$; (Middle) As a function of ρ, $t = 25.8x_{max}/d$ $(2ms)$; (Bottom) Dependence on t, with $\rho = 0.6$.

even at the highest ρ available to us. The Data pinning algorithm shows better performance than all the adaptive estimators. The composite estimator, using Data Pinning, is now the best performer

in all plots and for both the Sup and L1 measures. However we see that for large enough t, the difference between estimators begins to diminish as they are asked to deliver the impossible, with errors that correspondingly approach the maximum of 1, first in the Sup metric, and then L1.

6.6.2 Active-Passive Experiment

To test our estimators in real network conditions, we conducted novel experiments in which we sent probe streams along a link in a tier-1 ISP. Our experiments were novel because we simultaneously monitored the arrivals and departures to the hop. We described how we conducted the experiment in Chapter 3. Note that the cross-traffic was real Internet traffic and not controlled in any way. The utilization of the link was around 50%. The arrival and departure timestamps were accurate to sub-microsecond levels and allowed us to calculate the marginal of C. Since the arrival timestamps of cross-traffic to the output queue could not be measured, we could not calculate B. Hence we only evaluated the performance of our estimators in calculating C. Essentially, we performed 1-hop probing because, on a multi-hop path, queueing at hops other than the predominant bottleneck add noise to the observed delays. Quantifying the impact of such noise is path-dependent and out of our scope.

Due to probing load constraints we could only conduct $N = 10$ experiments at a particular time. We sent $n = 250$ packet pairs of size 40 bytes each, for a range of separations $t * d$ varying from the minimum possible 625 bytes ($t = 500ns$) to 40KB ($32\mu s$). Utilization levels could not be controlled in the experiments, however the performance of the estimators could be tested under operational conditions for a range of timescales t.

The resulting t-dependence performance curves are plotted in Figure 6.22. Under the realistic conditions of this experiment too, the results, at least for $td \geq 5$, are reminiscent of those seen earlier. In particular, the composite estimator performs the best, with an error at most half that of the commonly used $\mathbf{r} = 0$ estimator over a wide range of t. Also, as t increases, the performance worsens much much more slowly using the composite or $m = 8$ adaptive estimator than the Class 1 estimator with $\mathbf{r} = t - x$. The graphs show that estimates with Sup-norm (L1-norm) RMSE not

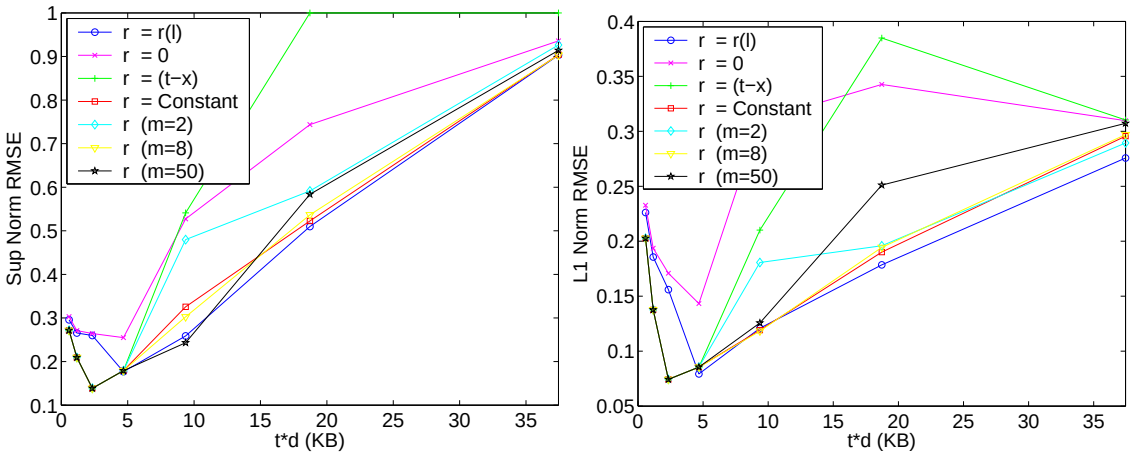

Figure 6.22. Estimator performance in live active/passive experiments. Dependence on t ($\rho = 0.50$).

exceeding 0.35 (0.15) can be obtained even for $t * d$ about 10 times the transmission time of $p_{max} = 1500$ bytes.

For very small t values near $1\mu s$, we see a clear *increase* in MSE. There are two possible reasons for this. First, we observed that errors in our probe generation times (essentially, our control of t) could be as large as 50% over these scales. Second, the independence assumption is much more likely to break down at these time scales.

6.7 Discussion

In this section, we discuss practical issues related to our analysis and estimators. First, we discuss how our estimators can be used in practice, especially without knowledge of absolute one-way delays. Then, we illustrate the power of our approach by demonstrating how it can be used to design practical methods with different trade-offs between accuracy and intrusiveness.

6.7.1 Practical Issues

So far, for simplicity, we have assumed synchronized sending and receiving times and no propagation delay. Neither of these assumptions is necessary since our estimators only use delay variation, $S - R$, of the excess delays R and S. In practice, the observed end-to-end delays, R' and S' can be

subtracted from their (observed) minimum values to obtain R, S pairs. Note that the resulting $S-R$ values are the true $S - R$ values. Accuracy of R is not necessary for calculating the conditional probabilities either. What is required is that two pairs with the same R value share the same true R value. This clearly holds when we subtract the minimum. Inaccurate absolute R values can cause us to inaccurately identify the beginning of the Class 1 region. Since our estimators do not use the boundary, such inaccuracies are also inconsequential.

Our estimators output the CDF and joint distribution of the cross-traffic functionals in time units (of service at the single hop). Converting this to hop utilization is easy. However, as with prior work [SKK03], converting this into units of bytes per second requires knowledge of the link capacity. For instance available bandwidth is the capacity multiplied with $1 - \rho$ where ρ is the utilization. Any of the well-known capacity estimation techniques, e.g., [Jac97, KCL$^+$04], can be used for this purpose. Naturally, inaccuracies in capacity estimation would affect available bandwidth estimation using our estimators.

6.7.2 Trade-off involving Intrusiveness and Accuracy

The inversion framework that we have used in the last two chapters provide new insights into making our proposed estimators as well as previous estimation techniques robust. We discuss this now.

Our work was motivated by prior works [RCR$^+$00, SKK03] that tried to ensure that a pair of probes share the same busy period by sending them very close to each other. By only choosing pairs in which the first probe saw a large relative delay, we ensured the same without requiring such small intra-pair separations. We assumed that R and (B, C) are independent so that this subset of pairs sees, on average, the same cross-traffic as all the pairs. Our assumption is redundant if we do not observe values of R below $t - x$, i.e., the two probes always share a busy period. This can be forced by sending back-to-back "filler probes" before the first packet of the pair. The filler probes would fill up the queue and ensure that the first packet of the pair saw large delays. The more filler probes we send, the higher is the probability of not observing R below $t - x$. This provides a desirable practical trade-off between intrusiveness and accuracy, i.e., if we send more filler probes we increase

intrusiveness while improving accuracy and requiring lesser use of our assumption. Thus, using the delays of the last two probes of a train is the "better" method of estimation than, for instance, IGI [HS03] which compares the delays of all consecutive packets of a train.

Filler probes can make fundamentally intrusive methods [HLM$^+$04, JD03, RRB$^+$03] robust too. These techniques send long packet trains and use "increasing delay trends" to infer the saturation of available bandwidth. They are forced to use parametrized heuristics to guard against false positives, i.e., perceived increasing delay trends even though the queue might have been idle between some packets of the train. Such techniques can also be made robust my sending filler probes before the train. These build up a large enough queue that the queue is very likely to be busy between the first and last packet of the probe train. The more filler probes, the higher is the probability of such "busy-ness". Hence, available bandwidth saturation can be tested by simply checking if the delay of the last packet of the train is larger than the delay of the first packet of the train.

6.8 Conclusions

In this section, we designed practical estimators based on the inversion expressions derived in the previous chapter. We encountered a variety of issues, related to estimating CDFs, such as monotonicity, bias-variance trade-offs. We proposed various estimators to tackle these issues. Using a combination of simulations, router traces obtained from detailed router monitoring experiments and novel live experiments with active injection and simultaneous passive monitoring, we compared the various estimators. We found that our estimators allow the intra-pair gap to be about 10 times the transmission time of the largest-sized packet while accurately estimating the entire CDF. Thus, we provided a generalization of packet pair methods in the sense that much more information can be extracted with far greater intra-pair separations than has been supposed in the past. Hence, it is much easier to use (non-intrusive) packet pair probing to paths with a single predominant bottleneck hop. Our methods also allow simple yet powerful extensions that trade-off intrusiveness for accuracy too, e.g., by sending an acceptable number of "filler" packets before each packet pair to increase the observed values of R.

Chapter 7

Architecture: Measurement-Friendly Networks

"Decide what you want, decide what you are willing to exchange for it. Establish your priorities and go to work."

- H. L. Hunt, American oil tycoon

The multi-hop nature of Internet paths has caused some of the most difficult problems, e.g., multi-hop queueing effects, faced in designing active measurements. Earlier in this dissertation, we encountered two specific examples of this. In Chapter 5, we found it hard to extend our single-hop cross-traffic inversion expressions to a multi-hop path. In Chapter 4, we discussed how our inability to control probe arrivals at intermediate hops makes it hard to sample those hops in an unbiased manner. The need to measure non-intrusively makes it even more difficult to deal with multiple hops. In this chapter, we seek to understand if and how these twin challenges of measuring a multi-hop path and non-intrusiveness can be tackled *assuming* that we can use newer architectural primitives. Since our focus is on active measurements, we discount architectural primitives that collect/share data (see [SSK97], for example) and only consider primitives based on packet forwarding. In the process, we also hope to understand the fundamental limitations of active measurements. We start by showing that each of the two challenges are naturally tackled if measurement packets are endowed with hop-dependent (high or low) priorities. This is because probes treated with high priority at a hop do

not observe any undesirable queueing at that hop. Additionally, probes treated with low priority at a hop are not only non-intrusive but also intrinsically measure the normal data queue because they are transmitted only when the latter is empty. We show that these network primitives do allow a wide range of active measurements to measure per-hop and end-to-end metrics. We also find that the *nonpreemptive* nature of packet forwarding and *persistence* of cross-traffic cause unavoidable inaccuracies and biases which can often be overcome though. Our work is important in three ways. First, it is useful in an immediate practical sense because, by enabling our primitives, network operators can actively measure their networks non-intrusively. Moreover, such a *Measurement-Friendly Network (MFN)* can be implemented using already-deployed router functionality in a low-overhead manner. Thus, our architecture is a cost-effective network management alternative to costly, passive data collection mechanisms. Second, we allow operators a way to expose the measurement facilities to end-users in a controlled manner. Our focus on non-intrusiveness along with easy-to-deploy access control mechanisms make it possible to do this in a "safe" manner. Third, our work provides little-known insights into fundamental limits of active measurements.

This chapter is organized as follows. In Section 7.1, we use prior work to illustrate how multi-hop paths have been measured and why it has proved to be hard. We also describe our goals and assumptions and provide an overview of our contributions in this chapter. In Section 7.2, we consider each of our two challenges - the multi-hop nature of paths and non-intrusiveness and describe how they can be individually tackled using forwarding primitives based on priority queueing. We use these primitives to describe our proposed architecture for a Measurement-Friendly Network. In Section 7.3, we describe three basic methods to measure delay-based metrics in our proposed MFN architecture. In Section 7.4, we describe the various active techniques to measure delay-based metrics in our proposed architecture. These include techniques to measure per-hop queueing, busy periods etc. We also briefly discuss how loss can be estimated. In Section 7.5, we discuss techniques to measure bandwidth-related metrics and their pros and cons. We discuss important issues such as packet marking and deployment overhead in Section 7.6. In Section 7.7, we summarize our contributions, namely, a cost-effective way by which network operators can (actively) measure and manage their networks accurately without costly, passive data collection mechanisms.

7.1 Overview

In this section, we survey prior approaches to tackling the multi-hop issue and intrusiveness. Using this, we motivate our target problem and state our assumptions. We end this section by providing an overview of our contributions.

7.1.1 Motivation

Active measurements can be used to measure either end-to-end properties, e.g., end-to-end delay, or per-hop properties, e.g., cross-traffic arrival process to a particular hop. Measurement of observable end-to-end properties such as delay and loss, pioneered by Bolot [Bol93] and Paxson [Pax99], involves issues mostly related to sampling, i.e., at what times do we send probe packets to obtain unbiased estimates. We thoroughly investigated this issue in Chapter 4. We saw, however, that arrival times of probe packets to intermediate hops cannot be controlled. Lack of such control also makes inversion hard - we saw this in Chapter 5 when we attempted to extend our single-hop cross-traffic inversion expressions to include the effects of multiple hops.

Previous work has also encountered difficulties stemming from the multi-hop nature of network paths. Most available bandwidth techniques (for example, [SKK03, JD03]) are fundamentally based on assuming a single (predominant) hop. Recent work [LRL05a] also showed how queueing at other hops affects these techniques. Moreover, this is reduced as probing intrusiveness increases. To our knowledge, TTL expiry is the only known way of measuring multiple hops of a path. For instance, *traceroute* [Jac87] uses ICMP replies to TTL-expired probe packets to determine the IP addresses of various hops along a path. *Pathchar* [Jac97] and its variants [LB00, PV02a] also use TTL expiry (and if necessary, the subsequent ICMP responses) to calculate individual hop capacities and queueing delays. Bottleneck-detection techniques [HLM$^+$04, ASM03] use the queue buildup ideas of available bandwidth techniques in conjunction with TTL expiry to estimate whether a particular link is a bottleneck or not. Though all these multi-hop techniques are quite sophisticated, they are not ideal for a variety of reasons. Those that rely on ICMP replies ignore effects related to reverse-path queueing and ICMP forwarding anomalies. The bottleneck-detection techniques are fundamentally intrusive because they rely on queue buildups and are restricted to providing

qualitative estimates. Also, it is unclear what kinds of inaccuracies these methods experience and why.

7.1.2 Scope

The above discussion illustrates three aspects of the current state-of-art on active measurements.

- The multi-hop nature of paths inherently poses a difficult challenge.
- The only known way to (partially) tackle this challenge are network primitives - TTL expiry and subsequent ICMP generation.
- Even after using these primitives, ensuring non-intrusiveness remains a challenge.

These motivate us to explore additional network primitives that, by tackling the twin challenges of multiple hops and non-intrusiveness, can be used to architect a *Measurement-Friendly Network (MFN)*, i.e.. a network whose per-hop and end-to-end properties can be easily measured in a non-intrusive manner. By explicitly designing a network to be measurement-friendly, we not only influence current and future network evolution but also wish to understand the fundamental limitations of active network measurements.

From the early days of the Internet, much research has been done on network architectures. A few prior works have proposed architectures for better network measurements. Seshan, et al. [SSK97] proposed SPAND, a network architecture that enables end-users to share performance measurements obtained by passively observing existing data flows. Varghese, et al. [VE03] proposed a couple of passive data collection functions at routers to simplify complicated measurement tasks such as traffic matrices. We are motivated by similar principles as them, namely, measurements can be vastly simplified by adding certain well-defined network functions. We differ from them, however, in that our primary focus is on active measurements and hence, we discount the use of data collected at routers (see [LMB01], for example) and instead, focus on forwarding primitives only. The primitives exploited so far have been TTL-expiry and ICMP generation. The original rationale behind having these primitives was network stability and error reporting. Mahajan, et al. [MSWA03] suggested augmenting the timestamping mechanisms in ICMP for better user-level

(fault diagnosis) measurements. Though such mechanisms are useful, they essentially provide access to data collected at routers. Also, they do not focus on estimating per-hop metrics and hence, do not answer questions related to unbiased sampling of individual hops.

Network architecture design necessarily assumes a willingness for change. In recent years, security problems and competitive concerns have caused network operators to make their networks less transparent by, for instance, filtering ICMP messages. Yet, MFN architectures providing more transparency are of practical interest if they allow network operators to better measure their own networks. Moreover, the push to less transparency has not been universal especially in educational networks. End-user access to better measurement primitives are of interest to such operators as long as they can control access to them and their use does not lead to more intrusive measurements. Also, our focus is on IP networks and we assume that all the queues are visible at the IP layer. Immunity from hidden Layer-2 links and multipath is preferable but not necessary. Also, by default, we assume that normal data traffic is scheduled according in a FIFO manner. We intend to study whether or not this is necessary.

The metrics that we consider are of interest to operators and/or end-users and are mostly motivated by prior work. As discussed earlier, delay is observable. Hence, the rare probing strategy outlined in Chapter 4 can be used to calculate end-to-end delay-based metrics. These include delay marginals, minimum delay, jitter and round trip times. Similar per-hop delay metrics include per-hop propagation delay, queueing delays and busy period durations. The latter refers to the duration of time for which a queue is continuously transmitting packets before it is idle. Along with utilization, it gives a sense of cross-traffic burstiness. Joint statistics of all these metrics are also desirable. The end-to-end loss metrics include loss episode frequencies and durations. The same metrics on a per-hop basis are also of interest.

The most basic bandwidth metric is link capacity. Available bandwidth has also received a lot of attention due to its applicability to a variety of situations including end-to-end congestion control, network health monitoring. We are interested in the per-hop and per-path capacity (so-called narrow link capacity) and available bandwidth. Utilization is obtained by subtracting, from 1, the available bandwidth divided by capacity. As with delays, we are interested in the entire statistics of available bandwidth. More detailed statistics include the cross-traffic characteristics (burstiness

may be measured by the functional B introduced in Section 5.2). Measuring finer metrics than the aggregate traffic such as flow size distributions are beyond our scope.

7.1.3 Contributions

To our knowledge, our work is the first to address the question of using network primitives to simplify active measurements. We show that hop-dependent priority queues are powerful enough to simplify many active measurements. We discuss a variety of techniques that can be used to measure end-to-end and per-hop properties with very low measurement overhead. Assuming that high-priority queueing is used as prescribed, these techniques can be considered non-intrusive. Such an assumption is not acceptable if entities other than the operator, e.g., end-users, are performing measurements. Our work provides two alternatives - the use of only low-priority queueing and access control mechanisms. The former exploit the little-known ability of low-priority packets to measure queues. The latter provides the network operator explicit access control mechanisms over the use of the MFN primitives. These access control mechanisms allow end-users to perform measurements while constraining them to be (almost) non-intrusive.

We also discuss how practical issues such as deployment of our MFN primitives. One of the disadvantages of our architecture is the presence of inaccuracies due to nonpreemption and cross-traffic persistence. We discuss why these can be thought to represent fundamental limitations of active measurements. We also show that, often, these inaccuracies can be ignored or eliminated.

7.2 Architecture Overview

In this section, we motivate and design our proposed Measurement-Friendly Network architecture. We start by considering the two challenges of multi-hop paths and non-intrusiveness in Section 7.2.1. We develop forwarding primitives for measurement packets that individually tackle each of these challenges. Our primitives are based on hop-dependent high and low priority queueing. In Section 7.2.2, we use these primitives to describe our proposed MFN architecture and discuss

important practical issues faced in using this architecture. In later sections, we design techniques to measure various metrics in MFNs and discuss the impact of these practical issues.

7.2.1 Design Motivation

We consider the multi-hop queueing effects and non-intrusiveness, one-by-one, and develop network primitives to tackle these two challenges. The challenges automatically motivate the use of priority queueing.

Hop Isolation Property

Consider the problem of estimating the queue sizes seen by normal data packets on the hops of a multi-hop path. Such per-hop congestion is essential to route delay-sensitive applications such as Voice-over-IP (VoIP). In the case of a single-hop path, it is trivial to measure the queue sizes. The end-to-end delay of a measurement packet is directly proportional to the queue size of the single hop. However, on a multi-hop path, the end-to-end delay is the sum of the queue sizes at all the hops. Hence, the single-hop technique based on end-to-end delay is not easily extended to multi-hop paths. Even if a multi-hop path contains a single predominant bottleneck (with much larger queues than the other hops), the noise from the other hops cannot be quantified.

The single-hop technique is applicable, though, if measurement packets experience the delay due to a single, specified hop h of the path. In other words, queueing due to normal data traffic at hops other than h should not delay measurement packets. This is possible if the measurement packets are treated with higher priority than data traffic at all hops but h. In other words, such hop-dependent priority queueing endows measurement packets with a *hop isolation property*.

Intrinsic Measurement Property

Consider the challenge of non-intrusive measurements. By definition, non-intrusiveness is achieved if the measurement packets do not cause the forwarding of normal data packets to be affected. This motivates a priority-queueing primitive opposite to the above primitive, namely, as-

signing lower priority to measurement packets. Low priority packets possess what we refer to as the *inherent measurement property*. We illustrate this using a simple example *ns-2* [Sim] simulation of a single hop that uses a strict 2-priority scheduling discipline. Data traffic has normal priority and only measurement packets have a low priority. The link uses a First-in First-out (FIFO) queue for each of these priorities. Since the link uses strict priority, a low priority packet is transmitted only when there is no normal priority packet. This is shown in Figure 7.1 where we plot the size of the normal-priority queue and the times at which low-priority measurement packets are transmitted on a 10Mbps link. The queue sizes were calculated using our Ground Truth Calculator (GTC) described in Chapter 3. We see that transmission times of the measurements packets correspond to the idle periods of the queue corresponding to the data traffic. Thus, the low priority packets inherently measure the "idleness" of the normal priority queue.

The inherent measurement property is useful in many ways. For instance, since low priority packets only use up the idle capacity of a link, their average output rate is never more than the available bandwidth on the link. Hence, a stream of low priority packets whose rate is greater than the available bandwidth on a link are non-intrusive (they do not affect existing data traffic) and yet, measure available bandwidth. Similarly, the transmission times reflect the times at which the link is idle and, as we shall see later, are also useful in characterizing the busy periods and other characteristics of cross-traffic.

7.2.2 Measurement-Friendly Network Architecture

The above discussion motivates us to design a MFN using hop-dependent low and high priority queueing. Data traffic is considered to have normal priority and hence, data packets are also called N-packets. Measurement packets may be treated with high, normal or low priority at *each* hop and the priority at different hops need not be the same. Thus, at each hop, a probe packet is a H-probe, N-probe or L-probe respectively. We use a string of letters from $\{L, H, N\}$ to denote the priorities of measurement packets along a path. The first letter indicates the default priority and uses the subscript d to denote this. The rest of the letters apply to individual hops, specified by a subscript, and represent exceptions to the default. For example, $H_d N_2 L_5$ denotes that a measurement packet has high priority at all hops except hop 2 and 5 where it has normal and low priority respectively.

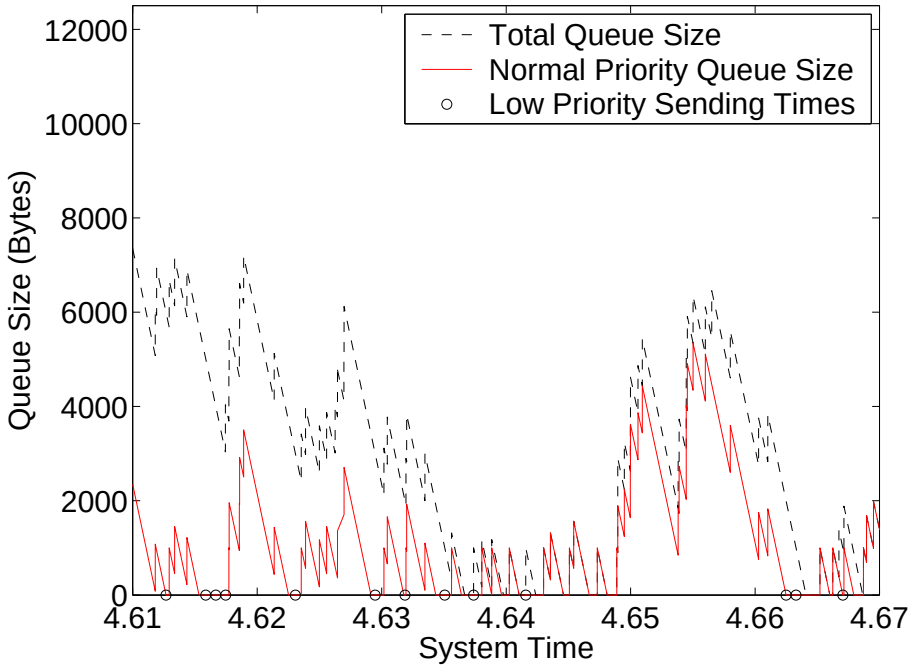

Figure 7.1. The normal-priority, total queue sizes and low-priority packet transmission times on a 10Mbps link. The low priority packets are only transmitted when there are no normal priority packets remaining.

Also, unless specified otherwise, we use priority queueing to refer to *strict* priority queueing, i.e., packet transmission is in the order H, N, L and packet dropping is in the order L, N, H.

From the next section onwards, we design techniques that use measurement packets with various combinations of hop-dependent priorities. We assume the ability to signal these priorities on a per-packet basis. We address practical signaling mechanisms in Section 7.6.2. For ease of exposition, we assume a single entity performing measurement. We develop techniques to measure a variety of per-hop and end-to-end metrics. For each metric, we first describe why it is useful and how it can be estimated in existing (FIFO-based) networks. Then, we describe a technique to estimate the metric in our MFN architecture. Using simple, illustrative simulations we evaluate our technique. With each technique, we repeatedly encounter one or both of the following issues that fundamentally limit the accuracy and correctness of measurement:

- **Nonpreemption:** So far, we implicitly assumed preemptive priority queueing, e.g., the transmission of an L-packet is paused when an N-packet arrives and an H-packet pauses the transmission of an N-packet (and L-packet). However, today's data networks are nonpreemptive.

A packet-in-transmission is never paused and hence, priorities are used only to select the *next* packet to transmit. This causes in-transmission L-packets to be intrusive by delaying normal data packets. Similarly, H-packets at a hop may experience "residual delays" due to packets-in-transmission that have low or normal priority.

- **Persistence:** We also find that persistence of cross-traffic on the path being measured causes dependencies between hops. Since the arrival of probe packets depends on prior hops due to nonpreemption, persistence can cause dependence between probing packets and what they measure. This dependence may lead to unbiased sampling.

In Section 7.6, we discuss the insights offered by our investigation into the inherent limitations of active measurements especially due to the above issues of nonpreemption and persistence. We also comment on important practical issues that need to be addressed for implementing our architecture, namely, how to achieve non-intrusiveness while using high priority queueing, deployment-related issues such as packet marking and router mechanisms and suitable access control methods. The latter is necessary when the network operator allows others to exploit the MFN primitives. We also discuss the implications of having multiple independent measurement streams.

7.3 Basic Methods

In this section, we investigate techniques to infer delay-based metrics in our MFN architecture. First, we consider sending individual measurement packets with high, normal and low priority at a target hop h and high priority everywhere else. We see that these packets can be used to estimate minimum delay (Section 7.3.1), queue size distributions (Section 7.3.2) and busy period statistics (Section 7.3.3) respectively. We find two sources of inaccuracies - nonpreemption and persistence. In Section 7.3.4, we discuss these and the maximum impact they can have. We find that, in most practical cases, error due to them can be tightly bounded.

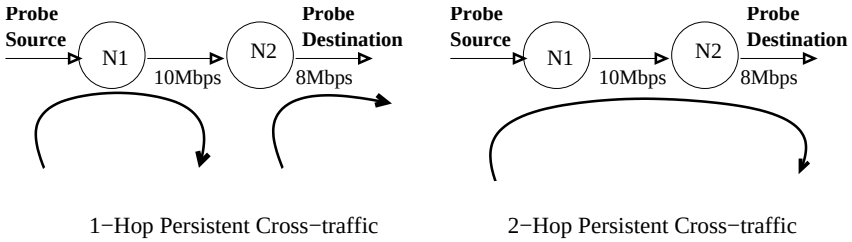

Figure 7.2. The two-hop systems we used had the same topology with cross-traffic having different persistence characteristics. In the topology on the left, cross-traffic persisted only for one hop whereas in the topology on the right, all cross-traffic on the second hop persisted from the first hop. In general, cross-traffic that is k-hop persistent traverses k consecutive links on the path.

7.3.1 Minimum End-to-End Delay

The minimum end-to-end delay along a path is the sum of the transmission and propagation times on all hops along the path. Since the transmission time is dependent on the packet size, the minimum delay is a size-dependent metric. It can be viewed as the best-case delay achievable along the path. Subtracting it from other metrics such as end-to-end delay makes it possible to distinguish congestion-related effects (due to queueing) from the physical limits of sending a packet from one physical location to another. In the current network architecture there are two ways of estimating the minimum delay. One way is to use estimates of capacities and propagation delay along all hops. Estimating propagation delay, however, requires timestamping functionality at all intermediate hops and, if absolute minimum delay needs to be estimated, clock synchronization. The minimum delay can also be estimated as the minimum delay observed by probe packets. Such an estimation method is accurate whenever at least one probe packet encounters no queueing delay at all hops. The probability of this occurring depends on the utilization of hops along the path. In the worst case, no probe packet would encounter empty queues if at least one of the hops is persistently congested.

In our proposed MFN architecture, the minimum end-to-end delay can be estimated by sending a H_d-probe, i.e., a probe packet treated with high priority on *all* hops. This probe packet, on account of its priority, would not see any queueing from normal data traffic. Recall that we are assuming a single measurement entity; Hence, such probe packets would not see any queueing from other H_d-probes as long as they are widely spaced. In Figure 7.3, we plot the CDF of delays experienced by 1000 such packets along the topology and cross-traffic characteristics shown in Figure 7.2. The cross-traffic we used consisted of three independent Poisson streams with packets of size 40, 576 and

1500 bytes. We used ns-2 [Sim] to implement these two-hop systems. As described in Chapter 3, we used in-built ns-2 functionality to implement priority queueing. In Figure 7.3, we make two important observations. First, not all packets observed the minimum delay. Second, even though the cross-traffic intensities in both topologies was the same, the CDF of probe packet delays changed.

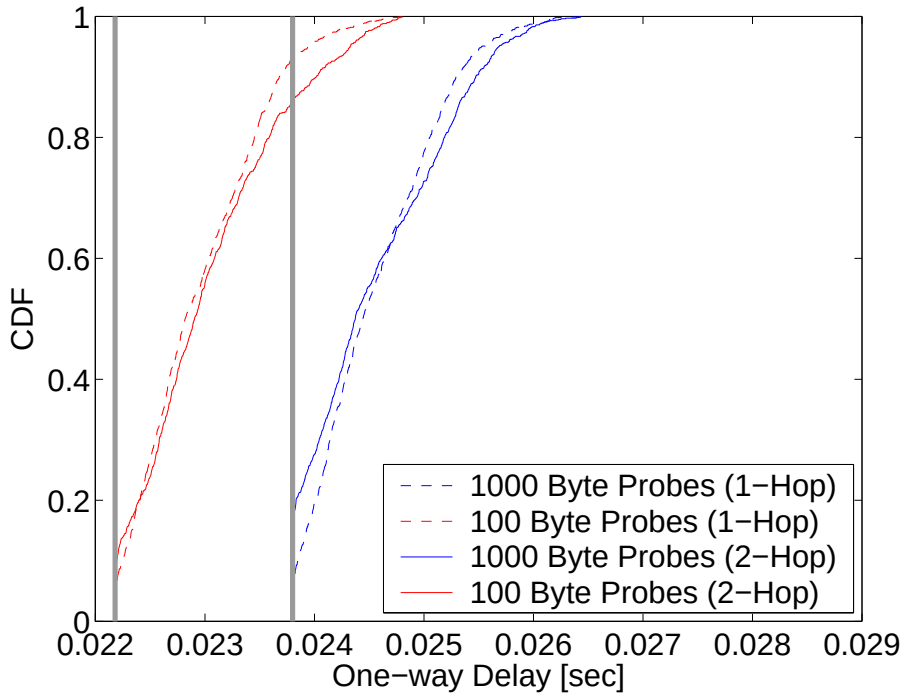

Figure 7.3. CDFs of end-to-end delay observed by 1000 H_d-probes of size 100 and 1000 bytes. For both sizes, we sent 1000 and 100 packets respectively though the average overhead was the same. The grey vertical bars represent the true minimum end-to-end delay. We also show whether cross-traffic was 1-hop or 2-hop persistent. Notice that the lack of preemption causes packets-in-transmission to add noise to the measured delays.

The reason why all packets did not observe the minimum delay is nonpreemption. At each hop, the current packet-in-transmission (if any) is completed before the H_d-probe is transmitted. This causes the observed probe delays to have a non-negative noise component, ϵ. On an n-hop path, this noise may be 0, the minimum, if the probe packet encounters no normal data packets on any hop. In the worst case, each probe packet may just arrive after the largest packet (of size p_{max}) at each hop. Hence,

$$\epsilon \in \left[0, \frac{p_{max}}{C_1} + \ldots + \frac{p_{max}}{C_n}\right]. \tag{7.1}$$

where C_i is the capacity of the i^{th} hop. We can remove the effects of the nonpreemption noise by sending multiple H_d-probes hoping that at least one of them would not encounter any packets-in-

transmission. Note that this can happen with a much higher probability that that of a probe packet experiencing no queueing in the current architecture. The number of probe packets to send, N, before one of them sees no nonpreemption noise is dependent on the probability distribution of ϵ. The probability distribution depends on various factors including cross-traffic packet sizes, arrival processes and cross-traffic persistence.

Cross-traffic persistence is the reason behind our second observation in Figure 7.3 that the two topologies observed different CDFs. Consider the system with 1-hop persistent cross-traffic. The cross-traffic arrival process at consecutive hops is independent. Barring exceptional situations, e.g., arrivals to consecutive hops are the result of multi-path forwarding from the same upstream hop, this is also true in practice. With independent arrivals, each hop h has no packet in its queue with an independent probability $1 - \rho_h$, where ρ_h is the utilization of that hop. The probability of seeing the minimum delay is equal to the probability of encountering empty queues all along the path and is given by $\Pi_h(1 - \rho_h)$. Therefore, the average number of probe packets that need to be sent before one of them experiences the true minimum delay is $1/\Pi_h(1 - \rho_h)$. The probe packet size does not affect these calculations. This is clearly seen in Figure 7.3 where the dashed lines just seem to be shifted horizontally (which corresponds to the difference in transmission times of the measurement packets).

However, in the case of 2-hop persistent cross-traffic, the queue sizes at the two hops are dependent on each other. This dependence causes ϵ, the noise due to nonpreemption, to be dependent on probe sizes. To see why, consider our simulation topology. Assume that the second hop has an empty queue and the first hop is transmitting a 576-byte data packet when a 100-byte H_d-probe arrives to the first hop. It is delayed by the in-transmission data packet at the first hop. By the time it arrives to the second hop, the 576-byte would be in-transmission there, too. In contrast, the second hop would have finished transmitting the 576-byte had we sent a 1000-byte H_d-probe. Thus, the inter-hop dependence causes the noise due to nonpreemption, ϵ, to vary based on probe size among other factors. This behavior is reflected in the different CDFs (even after accounting for the difference in transmission times) observed by 100 and 1000 byte packets in the 2-hop persistent path (see Figure 7.3). However, there is no single probe size that guarantees that, for all possible cross-traffic arrivals, the fewest number of probes need to be sent (on average) before one

of them experiences the minimum delay. Figure 7.3 shows that a small probe packet might be more likely that a large probe packet to encounter nonpreemption noise due to the same (large) persistent cross-traffic packet. But, this situation is reversed if the persistent cross-traffic packet encounters a small queue from other cross-traffic. In this case, a small probe packet, by arriving to the next hop quickly, can "overtake" the persistent cross-traffic packet that preempted it earlier. Hence, we recommend that the probes of the smallest possible size be sent with the singular goal of minimal probing overhead.

7.3.2 Per-hop Queueing

Congestion in a network is caused due to queue buildup. Pinpointing the congested hops is crucial to network operators who want to alleviate congestion and to end-user/overlays for rerouting traffic. In the current network architecture, for a single-hop, the delay observed by packets is directly proportional to the size of the queue at that hop. With multiple hops, the delay includes the sum of the queueing delays at all hops. Hence, there is no way to infer per-hop queueing delays just by observing end-to-end delays. With our MFN architecture, however, the queue size at any individual hop h can be determined by sending $H_d N_h$ packets, i.e., packets treated with normal priority *only* at hop h and treated with high priority on all other hops. This is a straightforward generalization of the single-hop method in the current network architecture.

In Figure 7.4, we plot the true and measured CDFs of queue sizes at the second hop of the topologies shown in Figure 7.2. We simulated these topologies using ns-2. We used $H_d N_2$-probes for measurements and subtracted the minimum end-to-end delay from their delays to calculate the queue sizes encountered at the second hop. We used our Ground Truth Calculator (GTC) to calculate the true distribution of queue size. We used Poisson cross-traffic and uniformly distributed probe which is justified by NIMASTA in Chapter 4. We find, in Figure 7.4, that the true and estimated CDFs differ from each other. Our earlier discussion indicates that nonpreemption might be the reason. To investigate this further, we plot the "noisy" versions of the true CDFs in Figure 7.4. These noisy versions are obtained by convolving the true CDF with the distribution of noise due to nonpreemption at the first hop. To make this convolution simple, we used cross-traffic with a constant packet size of 1500 bytes. Hence, nonpreemption noise is uniformly distributed between

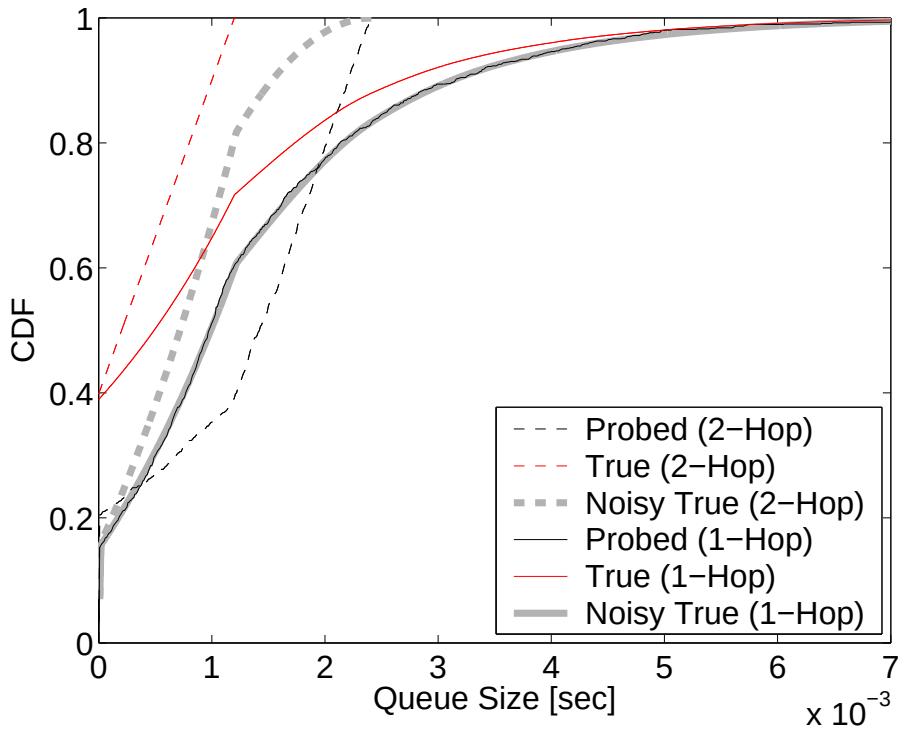

Figure 7.4. The true and ($H_d N_2$-probe) measured queue sizes at the second hop of the paths shown in Figure 7.2. The legend also shows whether the path had 1-hop or 2-hop persistent cross-traffic. We measure queue size in units of transmission time of the enqueued packets. The grey curves represent the true CDFs.

0 and the transmission time of a 1500-byte packet. Convolution of the true CDF with the CDF of the noise represents what we expect the probes to see if the noise is independent of the queue sizes encountered by the probes. We see that the noisy CDF does coincide with the measured CDF when cross-traffic is 1-hop persistent.

With 2-hop persistent cross-traffic, however, the noisy version does not coincide with what the probes saw. The reason is that, the noise is caused by an in-transmission data packet that is also enqueued before the probe at the second hop. This causes a dependence between the noise and the queue seen by the probe and hence, our calculated noisy CDF diverges from the CDF observed by probes. Based on the data and probe sizes, the probe packet can encounter the same data packet in transmission at all subsequent hops. Formally, if $Q_h()$ represents the time-dependent queue size of hop h, then we expect our $H_d N_h$ probes to reach hop h at times T_i. In practice, nonpreemption causes the probes to reach hop h at different times and also, adds a non-negative error to the observed

delay. Thus, we use the following approximation -

$$Q_h(T_i) \approx Q_h(T_i + \epsilon_{h,1}) + \epsilon_{h,1} + \epsilon_{h,2} \tag{7.2}$$

where $\epsilon_{h,1}$ is the nonpreemption noise until and excluding hop h and $\epsilon_{h,2}$ is the noise from hop $h+1$ onwards. Both these have a distribution similar to that in Equation 7.1. The $Q_h(.)$ terms will have the same distribution if cross-traffic is 1-hop persistent in which case only nonpreemption noise is an issue. With k-hop persistent cross-traffic, an *unavoidable sampling bias* results. Note that persistence *and* nonpreemption together cause this bias. A preemptive system with persistence would not have caused this bias and neither would a nonpreemptive system with no persistence.

7.3.3 Per-hop Busy Periods

Another indicator of congestion is busy periods at hops. These represent time intervals during which the queue is not empty. In the current network architecture, the best known way to estimate busy periods is to intelligently collect statistics at routers [HVPD04]. In our proposed MFN architecture, busy period statistics can be estimated using (low priority) L-probes which, we we discussed in Section 7.2.1, measure the "idleness" of queues. In a single-hop case, an L-probe would be transmitted only when the current busy period ends. This can be generalized to multi-hop paths by sending H_dL_h-probes. Note that such probes can only tell us when a busy period ended and not when it started. Hence, they measure the remaining duration of a busy period from an arbitrary point in time.

In Figure 7.5, we show the result of sending H_dL_2 probe packets for the topologies in Figure 7.2. We used our Ground Truth Calculator to obtain the true distribution of the remaining busy period duration. We plot the true CDFs, the noisy version of these CDFs and the estimated CDF. As before, we see nonpreemption noise and, in the case of 2-hop persistent cross-traffic, bias due to persistence and nonpreemption. Figure 7.5 confirms our previous results that, sampling is biased due to a combination of nonpreemption and persistence.

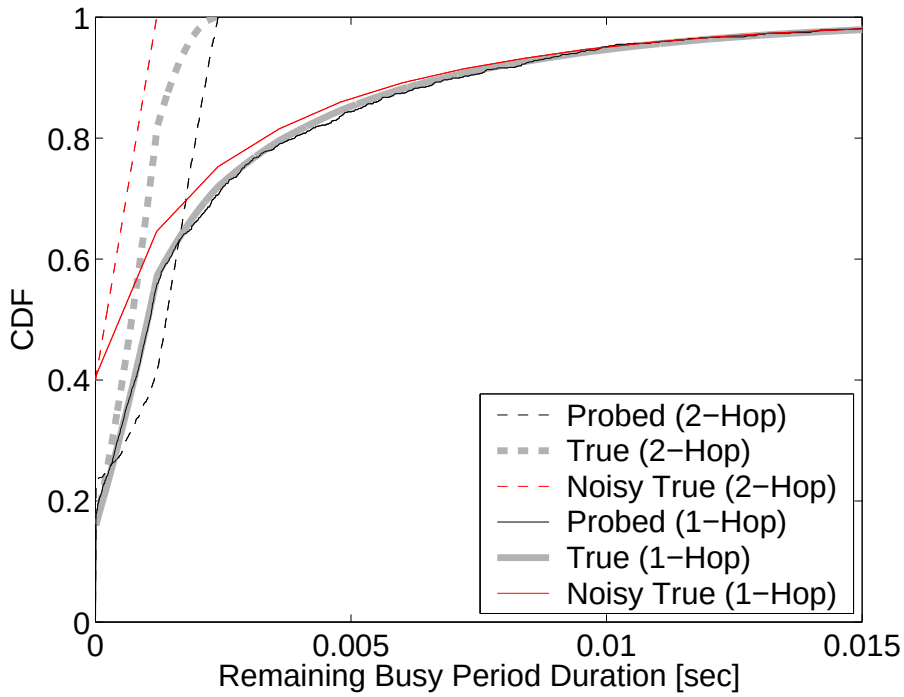

Figure 7.5. The true and measured (remaining) duration of busy periods from an arbitrary point in time. We also plot the noisy versions of the true CDFs obtained by convolving the true CDF with the distribution of nonpreemption noise.

7.3.4 Sources of Inaccuracies

We saw that noise due to nonpreemption (Equation 7.1) and sampling bias due to persistence and nonpreemption (Equation 7.2) cause inaccuracies in our proposed MFN architecture. In fact, these represent fundamental limitations of active measurements, i.e., when we intend to measure internal network characteristics using probe packets. Note that, both sources of inaccuracies are eliminated if the network is preemptive; Of course, this is hard to implement in a work-conserving manner.

How inaccurate do measurements become due to the above factors? First, consider noise due to nonpreemption. Equation 7.1 bounds the maximum value such noise can take. The upper bound is affected by the transmission times of the maximum-sized packets on all hops. This implies that noise is mostly due to the lowest-speed links on a path. For instance, on a path with a 100Mbps link and a 2.4Gbps link, the latter would contribute noise less than $5\mu s$, the transmission time of a 1500-byte packet. The disadvantage is that, to measure the high-speed link, the 100Mbps link

could cause noise up to $120\mu s$. However, as in Section 7.3.1, the delays of H_d-probes can be used to determine the cumulative noise from all hops. Such self-calibration is always desirable [Pax04]. Also, by deconvolving the cumulative noise from delays observed using H_dN_h-probes and H_dL_h-probes, we can make per-hop queue and busy period estimation accurate too. Note, however, that these estimation methods have noise from all but the target hop h whereas the cumulative noise includes all hops.

Sampling bias due to nonpreemption and persistence is given by Equation 7.2. By how much do $Q_h(T_i)$ and $Q_h(T_i + \epsilon_1)$ differ? It is hard to provide accurate bounds. Hence, we resort to intuitive arguments to motivate why the difference is small. Consider a two-hop path for which ϵ_1 is p_{max}/C_1. The only 2-hop persistent cross-traffic that can arrive at the second hop in $(T, T + \epsilon_1)$ is the in-transmission packet at the first hop. Thus, in this case, the difference between $Q_h(T_i)$ and $Q_h(T_i + \epsilon_1)$ is not more than the transmission time of a maximum-sized packet. Now, consider a three-hop path and assume that the probe packet got delayed by an in-transmission 3-hop persistent cross-traffic packet. If this persistent cross-traffic packet is enqueued at the second hop, the probe packet can get transmitted before this cross-traffic packet especially if it is small and arrives quickly to the second hop. Thus, in-transmission packets at one hop can be effectively preempted at a later hop and the bias need not accumulate.

7.4 Sophisticated Methods

In this section, we describe more sophisticated methods to measure delay and loss properties. In each case, we also address how the fundamental inaccuracies due to nonpreemption and persistence affect these methods. In Section 7.4.1, we present simple extensions, involving multiple probe packets, to the methods described in the previous section. In Section 7.4.2, we describe how joint statistics of queue sizes and busy periods can be estimated in our proposed MFN architecture. In Section 7.4.3, we investigate techniques that do not use high priority. We do so with the goal of developing techniques that end-users can exploit. In Section 7.4.4, we discuss how our architecture allows carefully-controlled non-intrusive loss measurements can be made.

7.4.1 Jitter

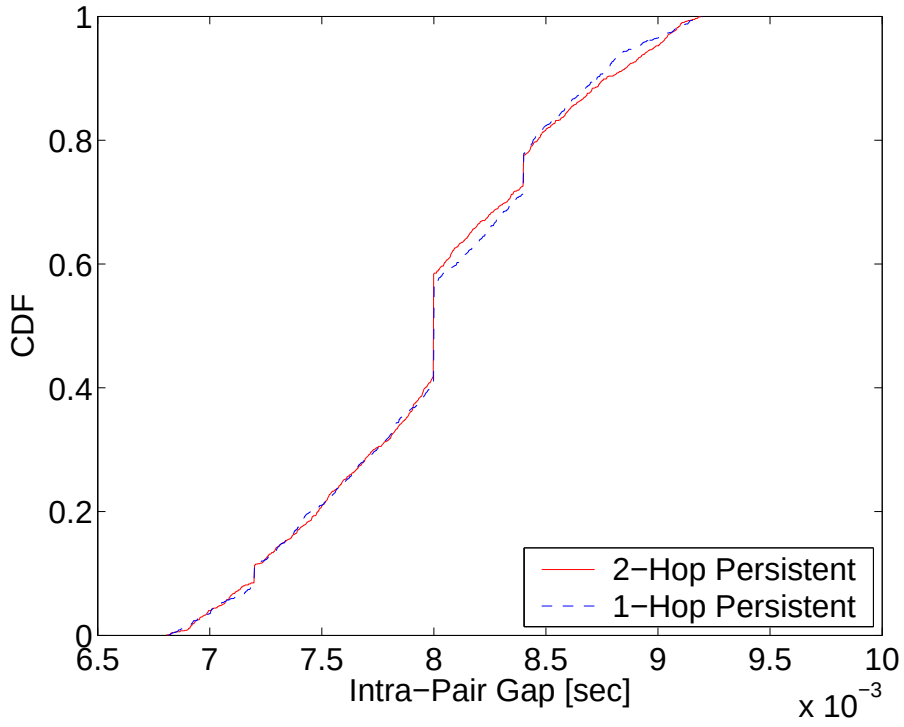

Figure 7.6. Simple plot illustrating that, when we send a pair of $H_d N_h$ probes, the intra-pair gap is distributed around the original gap t due to the noise due to nonpreemption. Since, we used two-hop systems, persistence does not affect arrival to the second hop. The difference between the two systems is normal estimation variance.

Techniques similar to those described in the previous section can be extended to probe pairs or trains as a way of estimating per-hop time-varying delay statistics, e.g., jitter. For instance, consider a pair of $H_d N_h$ probes with a time gap of t. Noise apart, their delays can be used to calculate the queue sizes at hop h t units apart. This is one possible measure of jitter and makes it easy to understand how the queue size at the target hop varies. As before, nonpreemption and persistence may cause the following approximation. We assume that, with preemption, the first probe would reach hop h at time T. Recall that $Q_h(.)$ denotes the queue size at hop h.

$$(Q_h(T), Q_h(T+t)) \approx (Q_h(T+\epsilon_1)+\epsilon_1+\epsilon_2, Q_h(T+t+\epsilon_1')+\epsilon_1'+\epsilon_2'). \tag{7.3}$$

Here, ϵ_1, ϵ_1' and ϵ_2, ϵ_2' represent the non-negative noise up to (and excluding) hop h and from hop $h+1$ onwards respectively. We illustrate the intra-pair gaps obtained with simulations of Figure 7.2

in Figure 7.6. This figure confirms that nonpreemption can cause the intra-pair gap to vary in an interval that is twice the transmission time of a 1500-byte packet, $2.4ms$.

7.4.2 Joint Statistics

Closely-sent probes can also be used to infer joint statistics across hops and of busy periods and queue sizes. For instance, consider a pair of equal-sized probe packets that are H_dN_h and H_dL_h. Also, let t be the gap between them, with the H_dN_h being sent first. We know, from our discussion on jitter above, that the gap between the packets when they reach hop h is $t \pm \epsilon_1$. The two probes measure the queue size and (remaining) busy period duration. Note that the second probe may have arrived at hop h during a newer busy period than when the first probe arrived. In any case, the busy period duration estimated from the second probe delay is an upper bound on the first probes' busy period duration. Due to the noise, we cannot make this a tight upper bound even by using $t = 0$. Notice that we can also send the H_dL_h followed by H_dN_h. Probe reordering would indicate that the busy period lasted more than t.

So far, we have seen that L-probes at a hop can be used to infer the remaining busy period duration. It turns out that probe packet triplets can be used to estimate busy period duration. The probes that need to be sent are H_dL_h, H_dL_h and H_dN_h in order with gaps of t_1 and t_2. The third packet, on account of being normal priority, can get sent ahead of the earlier probe(s). Thus, if it is received first, the busy period lasted longer than $t_1 + t_2$. If it is received second, the busy period lasted between t_2 and $t_1 + t_2$ time units. If no reordering occurs, the busy period lasted less than t_2 time units.

7.4.3 Non-Intrusive Techniques

The above discussion clearly illustrated the inherent measurement property of low priority probes. One possible issue with the techniques so far is that they require the probes to be high priority at all hops but h. This is not an issue if the measurements are being done by the network operator. Since we are also exploring if end-users can exploit these primitives, we investigate what can be done without high priority at any hop. Consider replacing the high-priority of the triplets,

discussed in Section 7.4.2, with normal priority, i.e., we send two $N_d L_h$-probes followed by a N_d-probe. The same arguments can be applied to infer busy period durations from reordering information with a crucial distinction. We cannot infer the duration of busy periods. Instead, we can determine whether or not two normal data packets sent with a time gap of t_1 or $t_1 + t_2$ would see the same busy period at hop h. Such information is crucial in determining how long congested periods at hops last - very useful information for user applications in coping with congestion.

Consider a non-intrusive stream of L_d-probes. The arrival times of these packets represent the total amount of time these probes had to wait for individual hops to experience idle periods. In a way, this provides a worst-case bound on the duration of busy periods of any hop.

7.4.4 Measuring Losses

Our proposed architecture also makes it easy to measure loss-related congestion. $H_d N_h$-probes measure the loss rates at hop h seen by normal data packets. In addition, receipt of $H_d L_h$-probes indicate the presence of idle periods at hop h and hence, the hop h is not continuously congested. This is useful information since continuous congestion is a much more unstable situation that is best avoided and could be the result of excessive network load, badly-provisioned router buffers, etc. In contrast, occasional losses are acceptable and indeed, common with TCP cross-traffic. $H_d L_h$-probes are advantageous also because they are non-intrusive at the target hop h, which could potentially be congested.

Our proposed MFN architecture also allows useful loss measurements without using high priority probes. Consider two simultaneous streams of $N_d L_h$-probes and N_d-probes. Losses among the latter indicate losses somewhere along the path. Losses among the former occur if some hop along the path has losses or hop h is continuously congested. Thus, discrepancy in the loss rates experienced by these probe streams would imply that hop h is a significantly congested hop. Moreover, this can be inferred using streams more non-intrusive than streams in the current network architecture. Sampling biases, resulting from cross-traffic persistence, could cause the two streams to observe different sets of network conditions. We believe that the non-intrusiveness of this technique makes it attractive even if biases are possible.

Note that, in the limit, loss measurements can start with L_d-probes and gradually make the probes normal priority at more and more hops. This ramp up from complete non-intrusiveness especially in congested network conditions has the attractive property of measuring the system while minimally affecting it. Losses of L_d-probes can also be used to estimate the starting window sizes of TCP especially in equation-based congestion control algorithms [FHPW00].

7.5 Bandwidth-based Metrics

As discussed in earlier chapters, active measurements have been used to estimate various bandwidth-related metrics such as link capacities, available bandwidth and cross-traffic. In the previous two chapters, we developed very useful cross-traffic estimators. Most of these bandwidth estimation techniques are essentially single-hop methods [JD04] that have been proposed for use with multi-hop paths. In this section, we discuss how our proposed MFN architecture aids such techniques. In Section 7.5.1, we discuss how existing capacity estimation techniques can be used. In Section 7.5.2, we discuss how our cross-traffic estimators are much more robust to noise caused by nonpreemption and persistence than other packet-pair methods which rely on fine-grained timing control [SKK03]. In Section 7.5.3, we discuss how we can measure time-varying bandwidth metrics such as available bandwidth, cross-traffic etc.

7.5.1 Capacities

Capacity estimation is primarily of interest to end-users. This is because link capacities change rarely enough that network operators can and do easily maintain databases with such information. Due to various overhead issues, the IP-level capacity is often lesser than the rated capacity of links (see [HVPD04], for example). Non-intrusive capacity estimation methods, therefore, provide information not easily available to operators. Two different kinds of capacity estimation techniques are known. The first kind send back-to-back packets assuming that they only see queueing at the narrow link. The time between the receipt of these two packets is assumed to reflect the capacity of the narrow link. Since these packets may encounter queueing, minimum filtering algorithms are used to select the pair that is likely to have seen no queueing. Capprobe [KCL+04] is an example

of such a technique. With our proposed primitives, sending H_d packets would eliminate all such queueing effects with the exception of nonpreemption noise.

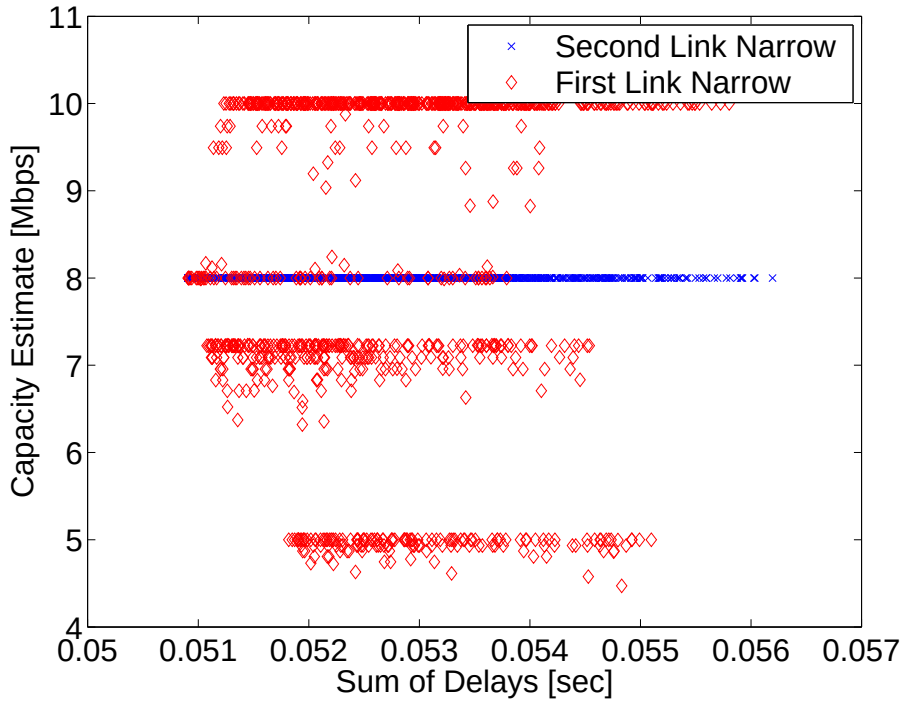

Figure 7.7. Plot showing the capacity estimates obtained using output dispersions on a two-hop path - the principle used in Capprobe [KCL$^+$04]. Two cases are shown - when the narrow hop is before and after the other hop. In the former case, the downstream hop alters the output dispersion. The minimum filtering proposed in Capprobe [KCL$^+$04] works as seen by the correct ($8Mbps$) estimate obtained with the smallest total sum. Nonpreemption is the main reason why minimum filtering continues to be necessary.

In Figure 7.7, we plot the capacity estimates obtained by using the principles on which Capprobe [KCL$^+$04] is based. We send back-to-back H_d-probes and estimate the capacity of the smallest (narrow) link using the size of these probes and the gap in their receiving times. We plot these estimates as a function of the total delays observed by both packets; Kapoor, et al. [KCL$^+$04] proposed using the pair with the minimum sum to calculate the capacity estimate. We make two observations from Figure 7.7. As prior works [PV02c, PV02a] showed, downstream hops can and do affect the output gap from the narrow link. This is seen in our MFN architecture because nonpreemption causes H_d-probes to also see residual queueing in the form of in-transmission packets. Our second observation from Figure 7.7 is that the minimum filtering algorithm does work. Moreover, if the second hop is narrow then nonpreemption effects are minimal.

A second kind of capacity estimation techniques, exemplified by *pathchar* [Jac97] and its variants [LB00, PV02a] use the transmission time of packets of various sizes to identify capacities of all links along a path. Each link is targeted using packets that expire at that links. Some variants [Jac97] rely on ICMP replies to such expired packets while other variants [LB00, PV02a] rely instead on effects such as packet tailgating. Again, nonpreemption and persistence could still add noise even if we try to eliminate queueing effects by using H_d-probes.

7.5.2 Estimating Cross-Traffic

In Chapters 5 and 6, we explored in detail how the delays of a pair of probe packets expose cross-traffic characteristics. We designed one-hop cross-traffic estimators that are immediately useful in our proposed architecture for measuring individual hops of an end-to-end path. For instance, sending $H_d N_h$ probe pairs would essentially cause the probe pairs to see queueing delays at hop h. In particular, Equation 7.3 tells us how accurately the delays of a probe pair reflect the across-hop delays. Recall that these inaccuracies result from non-preemption and persistence of cross-traffic. Our estimators are much more suitable in our proposed MFN architecture because we designed them with the explicit aim of not requiring them to be arrive close to each other at the target hop. In contrast, techniques such as Spruce [SKK03] require the packets to arrive at the hop at precisely-controlled times. This is not achievable even with $H_d N_h$-probes due to nonpreemption noise that is comparable to the intra-pair separations mandated by techniques such as Spruce [SKK03]. Our cross-traffic estimation techniques can also be combined with capacity estimates to calculate available bandwidth estimates of *individual hops* of a path.

7.5.3 Available Bandwidth

As discussed in Chapter 2, many fundamentally intrusive techniques to estimate available bandwidth have been proposed. These rely on saturating the available bandwidth and measuring the occurrence of saturation by observing the resulting (increasing) delay trends. Not only are these fundamentally intrusive they also cannot be used to estimate individual hops. We discussed in Sec-

tion 7.5.2 that, in our proposed architecture, non-intrusive cross-traffic estimation techniques can be used to estimate available bandwidth of individual hops.

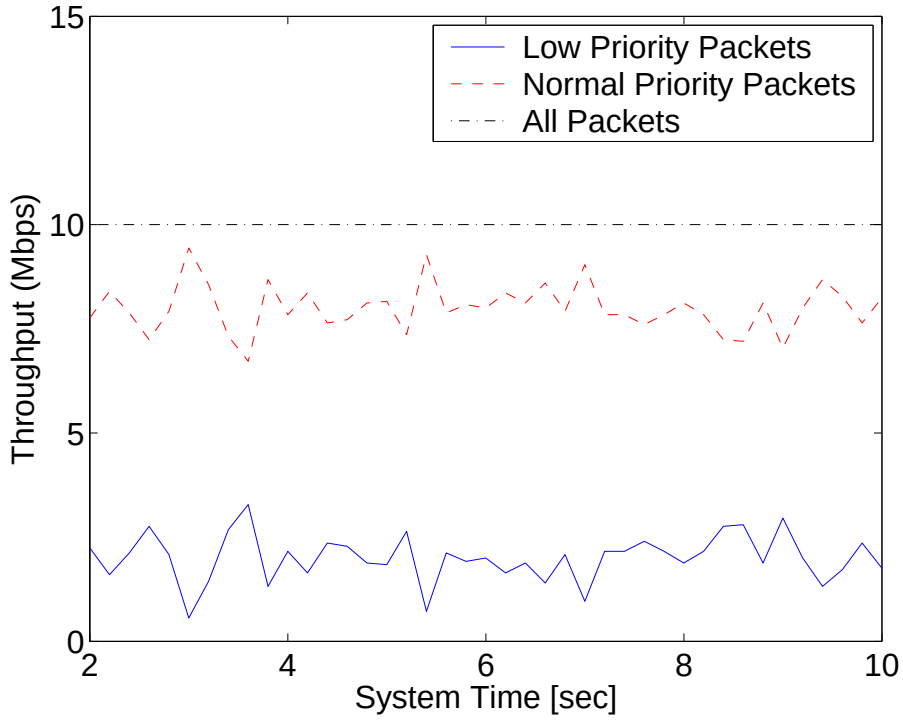

Figure 7.8. Plot showing the output rates of low-priority (probe) traffic and normal-priority (data) traffic from a single hop of capacity 10Mbps. The low-priority input rate was higher than the available bandwidth.

Our architecture lends itself to non-intrusive end-to-end available bandwidth estimation using only low-priority probe packets at each hop. To see why, we observe that low-priority packets at any hop capture the idle periods at a hop. Hence, the output rate of low-priority packets from a single hop is guaranteed to not be more than the available bandwidth. In fact, if the input rate of such low-priority packets is higher than the available bandwidth, their output rate would be exactly the available bandwidth (over some timescale). The advantage is they do not intrude on normal data packets. We show this in Figure 7.8 where we plot the input and output rates of normal and low-priority packets to a single 10Mbps link. We average the rates over 200ms intervals.

In Figure 7.9, we plot the output rates of L_d-probes in our canonical two-hop systems (Figure 7.2) with 1-hop and 2-hop persistent cross-traffic. Apart from variation in available bandwidth, we see little difference in the two cases. Clearly, persistence has no impact. Notice that nonpreemption effects are reversed in that they may impact the normal data packets. But, such impact is

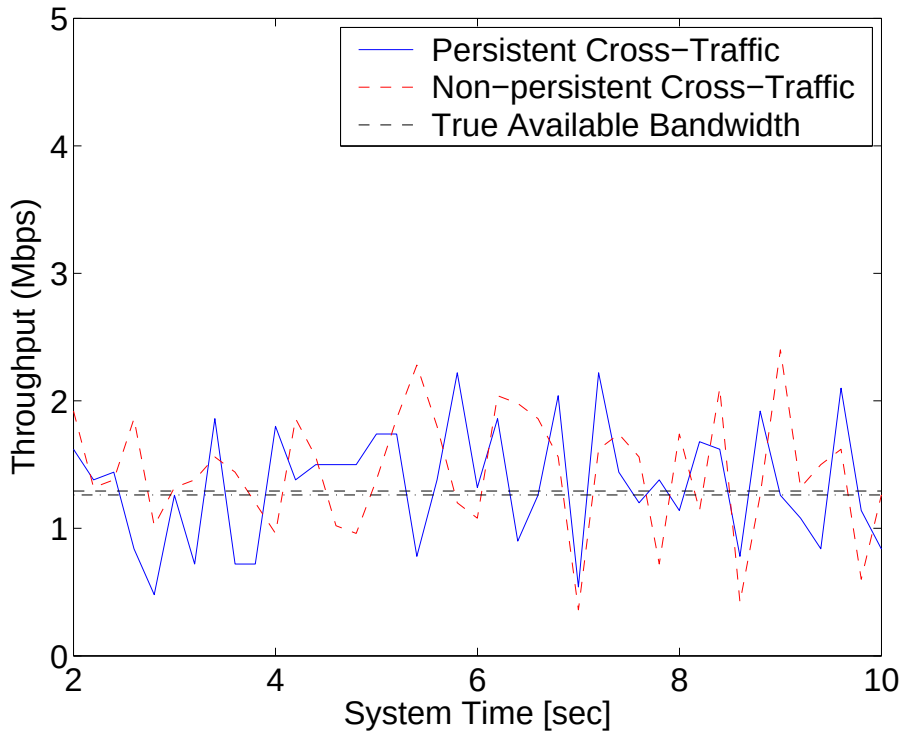

Figure 7.9.

very little because an in-transmission low-priority probe can delay normal data packets in the next busy period by a time that is at most its transmission time. Low-priority packets can also be used in congestion control algorithms such as TCP which essentially attempt to saturate the available bandwidth. For instance, TCP slow start ramps up the sending window (and hence, rate) to fill up the available bandwidth as quickly as possible. Slow start is an especially hard problem given the conflicting goals of fast estimation without being excessively intrusive by sending too much too fast. With low priority measurement packets, fast estimation is possible without intruding on existing data packets. Previous work [VKD02] has proposed the use of low-priority for large non-essential TCP transfers. They do not use low priority for any measurement purposes. However, recent work [SGF05] alluded to the use of low-priority TCP packets for similar reasons as above, namely, not intruding while increasing the sending rate.

Bottleneck location techniques [ASM03, HLM$^+$04] have adapted the principles of existing available bandwidth methods, namely, the observation of increasing delays. Using TTL expiry they saturate only a few of the hops. But, to not be excessively intrusive, they send probe trains

that are not very large. In our MFN architecture, non-intrusive L_d-probe trains can be as large as possible. Along with TTL expiry and increasing delays or even losses of L_d-probes, they can be used to even quantify the available bandwidth of choke links [HLM+04].

7.6 Discussion

In the previous sections, we saw how a variety of different measurements can be performed using probe packets with hop-dependent priorities. It is beyond our scope to define and compare specific estimators in our proposed MFN architecture. We now discuss important issues that arise in using our proposed architecture and measurement techniques in practice. In Section 7.6.1, we revisit the inaccuracy issues discussed earlier in Section 7.3. In Section 7.6.2, we discuss the necessary implementation issues for our architecture, how (probe) packets can indicate hop-dependent priorities and how priority queueing can be implemented. In Section 7.6.3, we discuss the intrusiveness of the various schemes and the implications on allowing end-users to use them especially high priority queueing. We also discuss appropriate access control mechanisms to restrict the effect of end-users performing measurements on normal data traffic. We conclude with a discussion on the pros and cons of our architecture in Section 7.6.5.

7.6.1 Hop Persistence and Dependence

As discussed in Section 7.3.4, existing data networks and routers implement nonpreemptive forwarding, i.e., a high-priority packet is delayed for a lower-priority packet that is currently being transmitted. We believe that the difficulty faced in building preemptive networks and routers makes it unlikely that they are going to be prevalent any time soon. However, nonpreemption introduces error that can be bounded and is mostly dependent (see Equation 7.1) on the transmission time of a maximum-sized packet on the narrow link of the path. We also saw nonpreemption and persistent cross-traffic causes inter-hop dependence and an unavoidable sampling bias. We also saw that the resulting sampling bias is likely bounded similar to nonpreemption noise. Not only is this noise relatively small but it can also often be eliminated. Furthermore, such noise is not relevant in techniques that do not use high priority. Though these sources of noise likely represent fundamental

limits of active measurements our work shows that their impact can be limited in our MFN architecture. We observe that, since the noise is inversely proportional to the link capacities on a path, its effect is much lesser within an ISP than end-to-end.

7.6.2 Deployment

Our techniques in the MFN architecture relied on the ability of probe packets to be treated with a potentially different priority at each hop. There are two deployment issues. One, how to implement the priority queueing. Two, how do probe packets indicate per-hop priority. The first issue, implementing priority queueing, requires only the necessary configuration options in current routers. This is because most routers today have implementations of Differentiated Services [BBC+98] that allow at least three strict priority queues to be defined. Thus, a network operator only needs to change the configuration of a router to treat appropriately classified probes with the suitable priority. Within an ISP, suitably configured filters at the edge of the network can prevent unauthorized use of the primitives especially high-priority queueing. Currently, we are exploring the feasibility of deploying our primitives in an operational network.

The second issue, that of probe packets indicating their priority, can be achieved in multiple ways. In the simplest case, network operators can just configure certain source/destination IP addresses and ports for each priority if measurements are only temporary. For a more permanent deployment too, network operators can use any combination of the Type-of-Service (ToS) and address fields of IP headers. A more systematic approach is desirable to prevent misconfiguration or if our proposed priority queueing primitives are exposed to end-users. Adding additional control information to IP headers has traditionally been difficult. IP options typically cause packets to take the so-called slow path instead of the fast path used by normal data packets. How many bits are required to implement all of our proposed functionality? One bit is needed to ensure backwards compatibility by treating existing data packets have normal priority. A ToS bit that is set to zero by default today can be chosen for this purpose. At each hop, we need to be able to specify three priorities. Assuming at most 15 hops on a path, there are 3^{15} combinations of priorities. They can be expressed in 17 bits. Given that measurement packets do not need arbitrary port numbers, we can leverage the 16-bit port numbers in addition to ToS bits. Notice that existing routers use port

numbers to classify packets and hence, such a scheme can be deployed today. Of course, if only a subset of priority combinations are used, much fewer bits are necessary. For instance, one ToS bit can be used to indicate L_d-probes.

7.6.3 Impact on Normal Data Traffic

One of our stated goals is to use non-intrusive measurements. An operator using our primitives for internal use controls the amount of probing load. All of our techniques which use high priority require very few probe packets. Moreover, none of them rely on large probe packet trains. Techniques which use low priority packets are, of course, non-intrusive and do not affect existing data traffic.

Opening up MFN primitives to end-users, if the operator is motivated to make the network more transparent, requires a combination of policy and access control. Policy limiting the amount of probing traffic is essential if end-users are allowed to use high priority. For instance, operators may rate-limit high priority at each hop to be less than 1% of the hop capacity. Since none of our techniques rely on sending many high-priority packets back-to-back, using a small queue for the high-priority packets is acceptable.

So far, we assumed that there is exactly one measurement stream. Some of our proposed techniques may have different interpretations in the presence of multiple independent measuring streams. For instance, high-priority probing packets from different measurement streams could cause (high-priority) queueing. Some methods, for example, the use of low priority packets to estimate available bandwidth, are robust to multiple measurement streams. In this particular case, other measurement streams can also be considered to be normal data traffic. There are two ways of solving the problem of multiple measurement streams. One way is to use a small queue for high priority packets. Thus, end-users can infer the presence of multiple measurement entities by observing losses. The second way is to have explicit access control mechanisms and filtering capabilities to control who can exploit our primitives.

7.6.4 Additional Metrics

A significantly more complicated metric is the traffic matrix [Sim]. Traditionally, it has been viewed as a network-wide metric. However, it can be viewed as being a function of cross-traffic rate and hop persistence. To understand this, consider 2 two-hop paths which share the first hop. The traffic matrix entry from the first hop to each of the second hops is given by the average cross-traffic rate multiplied by the hop persistence of this traffic, i.e., the fraction of cross-traffic that flows to the next link. Viewing it this way, traffic matrix estimation is also of interest to us. However, metrics such as flow durations are out of scope because these distributions can fundamentally change and still not affect the observed delays. To see this, simply mark every alternate packet of a flow as belonging to a new flow.

7.6.5 Pros and Cons

Our MFN architecture has many advantages. The primary advantage is that it helps achieve our goals of performing accurate active measurements non-intrusively without any additional data collection mechanisms. Moreover, it uses already-implemented priority queueing primitives and is arguably minimalistic. Though, we discussed only FIFO-based schedulers for normal data packets, our primitives can be applied even if this is not the case. The only requirement is that the low-priority and high-priority be with respect to *all* data packets. For instance, routers may treat data packets with two different priorities (which may be strict or not). As long as L-probes have lower priority than both kinds of data packets and H-probes have higher priority than both kinds, our primitives and techniques are applicable. This is especially relevant given that many access technologies have begun to use non-FIFO queueing disciplines [LPP04, WWZ$^+$05]. Our work also indicates how networks which implement priority queueing for data traffic can actually be measured much more easier than today's networks.

Two issues remain with our architecture. The first is the presence of inaccuracies due to non-preemption and persistence. We discussed why these inaccuracies can be bounded and often, be eliminated. Moreover, they appear to be fundamental in the sense that they are unlikely to be eliminated without making the network preemptive. It is an open question if other mechanisms can be

used to overcome these arguably small inaccuracies. The second important issue with our architecture is the use of high priority. We did show that, low priority alone is often useful especially for measuring end-to-end losses and available bandwidth. We also discussed rate limiting and access control mechanisms to limit the impact of high-priority probe packets. Whether or not better forwarding primitives can be designed is a question for future work that follows from our architecture. Finally, our architecture is not robust to anomalies such as link layer multiplexing and multi-path forwarding.

7.7 Conclusions

In this chapter, we focused on the question of designing forwarding primitives to tackle the challenge of non-intrusively measuring multi-hop path properties. We showed that hop-dependent priority queueing is an attractive solution to these challenges and can be used to design a Measurement-Friendly Network (MFN). We sketched a variety of measurement techniques to estimate end-to-end and per-hop metrics involving queueing delays, busy periods, losses, bandwidth and cross-traffic. We encountered nonpreemption and persistence as two barriers to accurate estimation. Though, these are unlikely to be eliminated without a radical change in the way routers are implemented, they are often small and can be ignored or eliminated. Thus, we provide a cost-effective, easy-to-deploy network management architecture for network operators - an alternative to costly, passive data collection mechanisms that typically require network-wide deployment. Moreover, if our proposed primitives are exposed to end-users, they can use the resulting transparency for better end-to-end performance and fault diagnosis. Network operators offering our primitives as a value-added service can thereby attract more customers. We discussed a variety of practical issues, such as access control, involved in enabling end-user access and how they can be addressed.

Chapter 8

Conclusion

"The average Ph.D. thesis is nothing but the transference of bones from one graveyard to another."

- James Frank Dobie, American folklorist

In this dissertation, we studied theoretical and practical questions concerning non-intrusive active network measurements. In this chapter, we conclude this dissertation with a summary of our work, open questions and potential for future work. This chapter is also organized like the dissertation. We first discuss sampling followed by inversion and end with architecture design.

8.1 Sampling

In Chapter 4, we studied the sampling-related aspects of active measurements, namely, how do we choose the times at which probe packets should be sent and why should we make this choice. Our starting point was the frequently-used "Poisson Arrivals See Time Averages (PASTA)" principle [Wol82]. PASTA has been used by prior work [Pax97a, Pax99] to justify Poisson probing as the only way of obtaining unbiased estimates.

Given our emphasis on non-intrusive probing, we started by investigating perfect non-intrusive probing to estimate the simplest metric, end-to-end delay. Perfectly non-intrusive probing implied that probes were of zero size. Though impractical, this allowed us to clearly understand the im-

plications of intrusiveness on sampling strategies. Using Palm Martingale Calculus [BB03], we rigorously analyzed the zero probe case. Based on this analysis, we formulated the *Non-Intrusive Mixing Arrivals See Time Averages (NIMASTA)* principle. This principle essentially states that, as long as *either* cross-traffic or probe arrivals satisfy the so-called mixing property, unbiased estimation is guaranteed. The mixing property is not restricted to Poisson streams and is satisfied by any stream with i.i.d inter-arrivals among others.

Upon analyzing the system assuming probes of non-zero size, we found that PASTA is the relevant principle. Thus, non-intrusiveness plays a key role in determining whether or not a sampling strategy has zero *sampling bias*. But, with intrusive probing, the bias is with respect to the "intruded-upon"(perturbed) system. Thus, Poisson probing leads to estimates that possess an additional *inversion bias*. Inversion bias needs to be removed to access the unperturbed system properties from our observations of probe delays in the perturbed system.

Both NIMASTA and PASTA are statements about asymptotic convergence, i.e., convergence to the ground truth assuming a large number of observations that tend to infinity. In practice, we are forced to work with a small number of samples and hence, estimation variance has a significant impact on estimation accuracy. Thus, in summary, the optimal sampling strategy depends on sampling bias, inversion bias and variance. Using simple one-hop simulations, we show that, with respect to a combined measure (root mean square error), non-Poisson streams perform better than Poisson streams. Furthermore, we motivated the use of a rare probing as a sound measurement strategy. The reason is that, probing that is rare enough can be considered to be non-intrusive and hence, have no inversion bias. NIMASTA is applicable and any mixing probe stream can be used in the absence of analytical tools that calculate the variance of probing streams.

We also showed that the use of PASTA to send probe pairs or trains at Poisson epochs is unjustified. Instead, appropriate extensions of NIMASTA are more suitable and again, sending pairs or trains rarely at mixing epochs is the sound strategy.

8.1.1 Future Work

Our work, while providing important insights into sampling-related issues, raises important open questions. We discuss them now.

The most important question that we left unanswered was the optimal sampling strategy to be used. Given that performance is a function of two different biases and variance, it is possible that the optimal strategy depends on the underlying cross-traffic properties. Works such as [Rou05] that calculate the variance of probing streams are steps in the right direction.

A question for future work concerning practical techniques relates to the rare probing strategy. We motivated the latter as a sound measurement strategy. We also alluded to good rules-of-thumb to determine how rare probing should be. More rigorous analysis is necessary to fully understand this question. In Chapter 4, we mentioned self-verifying tests, e.g., comparing the estimates obtained with probing streams of different degrees of rarity, as a possible solution. More investigation into the pros and cons of such tests is required.

8.2 Cross-traffic Estimation

In Chapters 5 and 6, we explored inversion problem related to accessing cross-traffic properties using the delays of a pair of probe packets. We summarize that work and discuss potential future work now.

8.2.1 Theory

First, we used Lindley's equation [Kle75c] to develop a packet-based analysis that relates the delays of a pair of probe packets to cross-traffic in the case of a single FIFO hop. We found that two functionals of cross-traffic, C and B representing the amount of cross-traffic and burstiness, relate the delays of the pair. Thus, they represent what we can hope to access using probe pair delays. Then, we showed that, without any assumption on cross-traffic, sample path ambiguity results, i.e., different cross-traffic functionals can lead to the same observed delays unless the intra-pair gap is

very small [SKK03]. In a sense, this represents the fundamental limits of probe pairs without any assumption on cross-traffic.

To investigate if the intra-pair gap can be increased to acceptable levels, we turned to inverting the functionals by making some assumption on cross-traffic. For maximum applicability, our goal was to not use any parametric model of cross-traffic but to use something more general. We assumed that the delay of the first probe packet is independent of the cross-traffic trapped between the pair. This assumption makes the system tractable. In fact, it allows the entire CDF of C and almost the entire joint density of C and B. We found that certain regions of the joint density, proportional to the probe size cannot be accessed and represent the cost of probing intrusiveness.

To access the cross-traffic functionals, we developed exact Class 1 expressions and approximate Class 2 expressions. The latter were developed with the view of performing inversion in the presence of very little data. We also tried to extend our single-hop analysis to a multi-hop path. We found that the inability to control timing at intermediate hops and cross-traffic persistence essentially make this a very challenging problem.

8.2.2 Practice

In Chapter 6, we adapted the inversion expressions developed in Chapter 5 for use with practical estimation. We started by defining three estimators that used a free parameter **r**. Using simple illustrative simulations that satisfied our assumption, we investigated the various factors affecting estimation. We found that, as utilization and/or burstiness reduces, the available data for estimation decreases. At very low utilization, estimation essentially becomes impossible. We also found a classic bias-variance trade-off operating that vividly demonstrated the utility of our Class 2 inversion expressions. This trade-off also illustrated the innovation of our approach - our ability to choose certain probe pairs, between which the queue is known to be busy, to perform estimation.

Then, we moved to considering CDF estimation and showed the importance of adapting **r**. We also showed how natural normalization of CDFs is desirable and using this, chose one of our three estimators as the best one to evaluate further. For this estimator, we investigated two ways of adaptive **r**. One of them was to use a saturation algorithm that, in loose terms, attempts to estimate

a so-called strong assumption curve reflecting the bias-variance trade-off. Since the resulting $\mathbf{r}$ does not estimate monotonic CDFs, we explored algorithms to enforce monotonicity. We also explored simpler constant $\mathbf{r}$ estimators.

We used a combination of simulations, router traces and live experiments to evaluate the various candidate estimators. We found that the saturation algorithm outperformed other estimators and the various monotonicity algorithms performed very similar to each other. In most of our experiments, we found that our estimators can estimate entire CDFs with an error less than 0.2 with an intra-pair separation about 10 times what previous works [SKK03] require when the hop utilization is 0.5 or more.

8.2.3 Future Work

We showed how our packet-based framework can be used to design techniques that trade-off non-intrusiveness for better accuracy using filler probes. Such methods that provide the ability to control intrusiveness are desirable and worthy of future work.

We used a well-motivated assumption on cross-traffic. It is desirable to investigate ways of verifying this assumption using observed probe pair delays. Investigating other such assumptions is another venue for future work. Also, we did not investigate the impact of probe size on estimation. It would be interesting to see if multiple probe sizes can be used to improve estimation.

8.3 Measurement-Friendly Networks

In Chapter 7, we investigated network primitives to tackle the twin challenges of non-intrusive active measurements that are robust to multi-hop queueing effects. The latter challenge has often been encountered by prior works and also by us, in Chapter 5. We showed how hop-dependent priority queueing can be viewed as a natural solution to these twin challenges. In particular, assigning probes high-priority at a hop makes the normal data queues at that hop invisible to probes. In addition, probe packets assigned low priority intrinsically measure the normal data queue. Thus,

well thought out combinations of per-hop priorities can be used to make a network measurement-friendly.

Using simple simulations, we designed active techniques to measure a multitude of per-hop and end-to-end properties in MFNs. The properties we considered included per-hop queueing delays, busy periods, available bandwidth, cross-traffic. The MFN architecture allowed the use of our 1-hop cross-traffic estimators on each hop of a path.

We also encountered unavoidable inaccuracies in MFNs. These are caused by nonpreemption and persistence of cross-traffic. We motivated why these represent fundamental limitations of active measurements. We also analyzed these sources of inaccuracies and provided lower and upper bounds on them. These showed that, the inaccuracies are mostly dependent on the transmission time of a maximum-size packet on the smallest link of the path. This, along with issues related to allowing end-users to use high-priority queueing, makes some aspects of our MFN architecture more suitable for use within an ISP.

We also discussed practical issues related to deploying our proposed architecture. It turns out our proposed primitives are already implemented in routers and only need to be turned on. We also discussed how the low-priority queueing primitives can be exposed to end-users without worrying about added intrusiveness. We also discussed access control mechanisms and rate limits that are necessary if end-users are allowed to exploit high-priority queueing at hops.

8.3.1 Future Work

Our work shows a novel use of architecture design, namely, to improve active measurement. An open question, in this context, is whether there are additional forwarding primitives that make active measurements easier. In particular, we required the use of high-priority queueing at certain hops. Can this be eliminated?

In the context of our MFN architecture, rigorous analysis to bound the inaccuracies due to nonpreemption and persistence is one avenue for future work. Such an analysis is helpful to network operators using our proposed architecture. Also, we briefly discussed various techniques that can be used in our MFN. Fine-tuning these techniques is necessary before they can be used in practice.

Bibliography

[ABKM01] David Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris. Resilient Overlay Networks. In *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*, October 2001.

[ANT01] Sara Alouf, Philippe Nain, and Don Towsley. Inferring Network Characteristics via Moment-based Estimators. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2001.

[AP99] Mark Allman and Vern Paxson. On Estimating End-to-End Network Path Properties. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, September 1999.

[ASM03] Aditya Akella, Srinivasan Seshan, and Bruce Maggs. An Empirical Evaluation of Wide-Area Internet Bottlenecks. In *Proceedings of Internet Measurement Conference (IMC)*, 2003.

[BB03] François Baccelli and Pierre Brémaud. *Elements of Queueing Theory*. Springer Verlag Applications of Mathematics, Second edition, 2003.

[BBC+98] Steven Blake, David L. Black, Mark A. Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. RFC 2475 - An Architecture for Differentiated Services, December 1998.

[BMVB06] François Baccelli, Sridhar Machiraju, Darryl Veitch, and Jean Bolot. The Role of PASTA in Network Measurements. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2006.

[Bol93] Jean-Chrysostome Bolot. End-to-end Packet Delay and Loss Behavior in the Internet. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, San Francisco, CA, September 1993.

[BOP94] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 1994.

[Bru71] Shelby L. Brumelle. On the Relation Between Customer and Time Averages in Queues. *Journal of Applied Probability*, 8(3):508–520, September 1971.

[CAI] CAIDA. Cooperative Association for Internet Data Analaysis. `http://www.caida.org`.

[CC96] Robert Carter and Mark Crovella. Measuring Bottleneck Link Speed in Packet-Switched Networks. *Performance Evaluation*, October 1996.

[CLZ87] David D. Clark, Mark L. Lambert, and Lixia Zhang. NETBLT: A High Throughput Transport Protocol. *ACM SIGCOMM Computer Communications Review (CCR)*, 17(5):353–359, 1987.

[Cox84] David R. Cox. *Long-range Dependence: A Review* Edited by H.A. David and H.T. David, pages 55–74. Iowa State University Press Ames (IA), 1984.

[Des67] A. Descloux. On the Validity of a Particular Subscriber's View. In *Proceedings of Fifth International Teletraffic Congress*, New York NY, 1967.

[DKS89] Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and Simulation of a Fair Queueing Algorithm. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, pages 1–12, 1989.

[Dow] Allen B. Downey. Clink. `http://allendowney.com/research/clink/`.

[Dow99] Allen B. Downey. Using Pathchar to Estimate Internet Link Characteristics. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 1999.

[DRM01] Constantinos Dovrolis, Parameswaran Ramanathan, and David Moore. What do Packet Dispersion Techniques Measure? In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2001.

[Dur96] Richard Durrett. *Probability: Theory and Examples*. Duxbury Press, second edition, 1996.

[DVJ88] D.J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes*. Springer-Verlag, 1988.

[End] Endace. `http://www.endace.com`.

[FHPW00] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer. Equation-Based Congestion Control for Unicast Applications. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2000.

[GL80] D.P. Gaver and P.A. Lewis. First-Order Autoregressive Gamma Sequences and Point Processes. *Advances in Applied Probability*, 12:727–745, 1980.

[Gla91] Paul Glasserman. *Gradient Estimation via Perturbation Analysis*. Kluwer Academic Publishers, Boston, 1991.

[HLM+04] Ningning Hu, Li (Erran) Li, Z. Morley Mao, Peter Steenkiste, and Jia Wang. Locating Internet Bottlenecks: Algorithms Measurements and Implications. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2004.

[HS03] Ningning Hu and Peter Steenkiste. Evaluation and Characterization of Available Bandwidth Probing Techniques. *IEEE Journal on Selected Areas in Communications (JSAC)*, August 2003.

[HVA03] Nicolas Hohn, Darryl Veitch, and Patrice Abry. Cluster processes a natural language for network traffic. *IEEE Transactions on Signal Processing, Special issue on Signal Processing in Networking*, 51(8):2229–2244, August 2003.

[HVPD04] Nicolas Hohn, Darryl Veitch, Konstantina Papagiannaki, and Christophe Diot. Bridging Router Performance and Queuing Theory. In *Proceedings of ACM International*

Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), pages 355–366, 12–16 June 2004.

[IEP] IEPM. Internet End-to-end Performance Monitoring. `http://www-iepm.slac.stanford.edu/`.

[IPP] IPPM. IETF IP Performance Metrics. `http://www.ietf.org/html.charters/ippm-charter.html`.

[Ixi] Ixia. `http://www.ixiacom.com`.

[Jac87] Van Jacobson. *traceroute*, 1987. `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`.

[Jac88] Van Jacobson. Congestion Avoidance and Control. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, August 1988.

[Jac97] Van Jacobson. Pathchar – A Tool to Infer Characteristics of Internet Paths, 1997. Presented at the Mathematical Sciences Research Institute (MSRI).

[Jai89] Raj Jain. A Delay-based Approach for Congestion Avoidance in Interconnected Heterogeneous Computer Networks. 1989.

[JD03] Manish Jain and Constantinos Dovrolis. End-to-end Available Bandwidth: Measurement Methodology Dynamics and Relation with TCP Throughput. *IEEE/ACM Transactions on Networking (TON)*, 11(4):537–549, 2003.

[JD04] Manish Jain and Constantinos Dovrolis. Ten Fallacies and Pitfalls on End-to-End Available Bandwidth Estimation. In *Proceedings of Internet Measurement Conference (IMC)*, 2004.

[KCL+04] Rohit Kapoor, Ling-Jyh Chen, Li Lao, Mario Gerla, and M. Y. Sanadidi. CapProbe: A Simple and Accurate Capacity Estimation Technique. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2004.

[Kes95] Srinivasan Keshav. Packet-Pair Flow Control. *IEEE/ACM Transactions on Networking (TON)*, February 1995.

[KK01] Aleksandar Kuzmanovic and Edward W. Knightly. Measuring Service in Multi-Class Networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2001.

[Kle75a] Leonard Kleinrock. *Queueing Systems*, volume II: Computer Applications. John Wiley and Sons, 1975.

[Kle75b] Leonard Kleinrock. *Queueing Systems*, volume I: Theory. John Wiley and Sons, 1975.

[Kle75c] Leonard Kleinrock. *Queueing Systems*, volume I (Theory) and II(Computer Applications). John Wiley and Sons, 1975.

[LB99] Kevin Lai and Mary Baker. Measuring Bandwidth. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1999.

[LB00] Kevin Lai and Mary Baker. Measuring Link Bandwidths Using a Deterministic Model of Packet Delay. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2000.

[LMB01] Matthew J. Luckie, Anthony J. Mcgregor, and Hans-Werner Braun. Towards Improbing Packet Probing Techniques. In *Proceedings of Internet Measurement Workshop (IMW)*, November 2001.

[LPP04] Karthik Lakshminarayanan, Venkata N. Padmanabhan, and Jitendra Padhye. Bandwidth Estimation in Broadband Access Networks. In *Proceedings of Internet Measurement Conference (IMC)*, 2004.

[LRL05a] Xiliang Liu, Kaliappa Ravindran, and Dmitri Loguinov. Multi-Hop Probing Asymptotics in Available Bandwidth Estimation: Stochastic Analysis. In *Proceedings of Internet Measurement Conference (IMC)*, October 2005.

[LRL05b] Xiliang Liu, Kaliappa Ravindran, and Dmitri Loguinov. What Signals do Packet Pair Dispersions Carry? In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2005.

[LRLL04] Xiliang Liu, Kaliappa Ravindran, Benyuan Liu, and Dmitri Loguinov. Single-Hop Probing Asymptotics in Available Bandwidth Estimation: Sample-Path Analysis. In *Proceedings of Internet Measurement Conference (IMC)*, October 2004.

[MA] Matt Mathis and Mark Allman. RFC 3148: A Framework for Defining Empirical Bulk Transfer Capacity Metrics, July. 2001.

[MBG00] Bob Melander, Mats Bjorkman, and Per Gunningberg. A New End-to-end Probing and Analysis Method for Estimating Bandwidth Bottlenecks. In *Proceedings of the Global Internet Symposium*, San Francisco, November 2000.

[MNA] NLANR MNA. Measurement and Network Analysis. `http://moat.nlanr.net/`.

[MSWA03] Ratul Mahajan, Neil Spring, David Wetherall, and Thomas Anderson. User-level Internet Path Diagnosis. In *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*, 2003.

[Muu83] Mike Muuss. *ping*, 1983. `http://ftp.arl.mil/~mike/ping.html`.

[MVB+05] Sridhar Machiraju, Darryl Veitch, François Baccelli, Antonio Nucci, and Jean Bolot. Theory and Practice of Cross-Traffic Estimation Poster. In *Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, June 2005.

[MVBB05] Sridhar Machiraju, Darryl Veitch, François Baccelli, and Jean Bolot. Theory and Practice of Cross-traffic Estimation via Probes. Technical Report RR-5763, INRIA-ENS, 2005.

[MW90] Benjamin Melamed and Ward Whitt. On Arrivals That See Time Averages. *Operations Research*, 38(1):156–172, 1990.

[MY95] Benjamin Melamed and David D. Yao. *Queueing: Theory, Methods and Open Problems*, chapter 7, The ASTA Property, pages 195–224. CRC Press, 1995.

[PÓ1] Attila Pásztor. psim, 2001. http://www.cubinlab.ee.mu.oz.au/probing/software.shtml.

[PAM00] Vern Paxson, Andrew Adams, and Matt Mathis. Experiences with NIMI. In *Proceedings of Passive and Active Measurement (PAM) Workshop*, 2000.

[PAMM98] Vern Paxson, Guy Almes, Jamshid Madhavi, and Matt Mathis. RFC 2330 - Framework for IP Performance Metrics, May 1998.

[Pat97] Pathchar. ftp://ftp.ee.lbl.gov/pathchar, 1997.

[Pax97a] Vern Paxson. End-to-End Routing Behavior in the Internet. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 1997.

[Pax97b] Vern Paxson. *Measurements and Analaysis of End-to-End Internet Dynamics*. PhD thesis, University of California at Berkeley, 1997.

[Pax99] Vern Paxson. End-to-End Internet Packet Dynamics. *IEEE/ACM Transactions on Networking (TON)*, 7(3):277–292, 1999.

[Pax04] Vern Paxson. Strategies for Sound Internet Measurement. In *Proceedings of Internet Measurement Conference (IMC)*, 2004.

[Pch] Pchar. http://www.employees.org/bmah/Software/pchar/.

[Pet83] Karl Petersen. *Ergodic Theory*. Cambridge University Press Cambridge England, 1983.

[PG93] Abhay K. Parekh and Robert G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-node Case. *IEEE/ACM Transactions on Networking (TON)*, 1(3):344–357, 1993.

[Pla] PlanetLab. http://www.planet-lab.org.

[PMAM98] Vern Paxson, Jamshid Mahdavi, Andrew Adams, and Matt Mathis. An Architecture for Large-scale Internet Measurement. In *IEEE Communications*, August 1998.

[PV02a] Attila Pásztor and Darryl Veitch. Active Probing using Packet Quartets. In *Proceedings of Internet Measurement Workshop (IMW)*, 2002.

[PV02b] Attila Pásztor and Darryl Veitch. On the Scope of End-to-End Probing Methods. *IEEE Communication Letters*, 6(11), November 2002.

[PV02c] Attila Pásztor and Darryl Veitch. The Packet Size Dependence of Packet Pair Like Methods. In *Proceedings of IEEE International Workshop on Quality of Service (IWQoS)*, 2002.

[RCR+00] Vinay Ribeiro, Mark Coates, Rudolf Riedi, Shriram Sarvotham, Brent Hendricks, and Richard Baraniuk. Multifractal Cross-Traffic Estimation. In *Proceedings of the ITC Specialist Seminar on IP Traffic Measurement Modelling and Management*, pages 1–10, Monterey, CA, 2000.

[Rou05] Matthew Roughan. Fundamental Bounds on the Accuracy of Network Measurements. In *Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, June 2005.

[RRB+03] Vinay J. Ribeiro, Rudolf H. Riedi, Richard G. Baraniuk, Jiri Navratil, and Les Cottrell. pathChirp: Efficient Available Bandwidth Estimation for Network Paths. In *Proceedings of Passive and Active Measurement (PAM) Workshop*, 2003.

[SAM] SAMI. Secure and Accountable Measurement Infrastructure. `http://www.psc.edu/networking/projects/sami/`.

[SB93] Dheeraj Sanghi and Suman Bannerjee. *NetDyn*, 1993. `http://www.cs.umd.edu/~suman/netdyn/index.html`.

[SBDR05] Joel Sommers, Paul Barford, Nick Duffield, and Amos Ron. Improving Accuracy in End-to-End Loss Measurement. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2005.

[SGA93] Dheeraj Sanghi, Olafur Gudmundsson, and Ashok K. Agrawala. Study of Network Dynamics. In *Selected papers of the 4th joint conference on European networking conference*, pages 371–378, 1993.

[SGF05] Manpreet Singh, Saikat Guha, and Paul Francis. Utilizing Spare Network Bandwidth to Improve TCP Performance. In *Poster in Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 2005.

[Sim] NS-2 (Network Simulator). `http://www.isi.edu/nsnam/ns/`.

[SKK03] Jacob Strauss, Dina Katabi, and Frans Kaashoek. A Measurement Study of Available Bandwidth Estimation Tools. In *Proceedings of Internet Measurement Conference (IMC)*, pages 39–44, 2003.

[SM98] Vinod Sharma and Ravi Mazumdar. Estimating traffic parameters in queueing systems with local information. *Performance evaluation*, 32:217–230, 1998.

[Spr] Sprintlabs. `http://ipmon.sprintlabs.com`.

[SSK97] Srinivasan Seshan, Mark Stemm, and Randy H. Katz. SPAND: Shared Passive Network Discovery. In *USENIX Symposium on Internet Technologies and Systems*, December 1997.

[Sto95] Charles J. Stone. *A Course in Probability and Statistics*. Duxbury Press, 1995.

[TDDA05] Muhammad Mukarram Bin Tariq, Amogh Dhamdhere, Constantinos Dovrolis, and Mostafa Ammar. Poisson versus Periodic Path Probing (or Does PASTA Matter)? In *Proceedings of Internet Measurement Conference (IMC)*, pages 119–124, Berkeley CA, Oct 2005.

[Tra] Traceroute.org. `http://www.traceroute.org`.

[TYNB04] Yolanda Tsang, Mehmet Yildiz, Robert Nowak, and Paul Barford. Network Radar: Tomography from Round Trip Time Measurement. In *Proceedings of Internet Measurement Conference (IMC)*, pages 175–180, Taormina Sicily Italy, Oct 2004.

[VE03] George Varghese and Christian Estan. The Measurement Manifesto. In *Proceedings of Workshop on Hot Topics in Networks (HotNets)*, 2003.

[Vei05] Darryl Veitch. Sampling and Inversion. In *Talk at LIP6 Sampling Workshop*, 2005.

[VKD02] Arun Venkataramani, Ravi Kokku, and Mike Dahlin. TCP-Nice: A Mechanism for Background Transfers. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[Wol82] Ronald W. Wolff. Poisson Arrivals see Time Averages. *Operations Research*, 30(2):223–231, 1982.

[WWZ+05] Wei Wei, Bing Wang, Chun Zhang, Jim Kurose, and Don Towsley. Classification of Access Network Types :Ethernet Wireless LAN ADSL Cable Modem or Dialup? In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2005.

[WZJ03] Zhiheng Wang, Amgad Zeitoun, and Sugih Jamin. Challenges and Lessons Learned in Measuring Path RTT for Proximity-based Applications. In *Proceedings of Passive and Active Measurement (PAM) Workshop*, 2003.

[ZDP01] Yin Zhang, Nick Duffield, and Vern Paxson. On the Constancy of Internet Path Properties. In *Proceedings of Internet Measurement Workshop (IMW)*, 2001.